CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Engineering

Subject Name: Object Oriented Programming with C++
Semester: II
Subject Code: CE144
Academic year: 2022-23

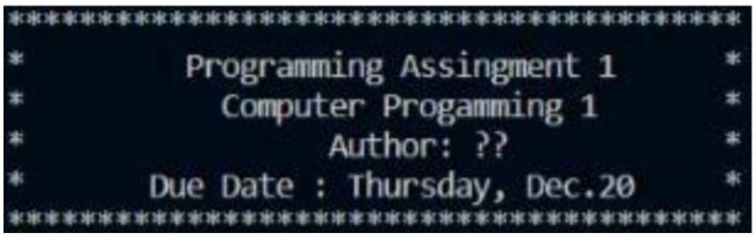
Practical Set- 1

Practical-1.1

Write a C++ program that will print output in the following form. Make sure your output looks exactly as shown here (including spacing, line breaks, punctuation, and the title and author).

Note: Use cout objects and endl manipulator.

Expected Output:



```
****************************************
*          Programming Assignment 1    *
*           Computer Progamming 1      *
*                Author: ??            *
*         Due Date : Thursday, Dec.20  *
****************************************
```

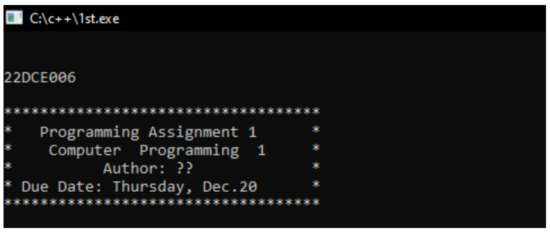
Attach the screenshot of the output.

Code:

```
#include<iostream>
using namespace std;
main()
{
   cout<<"22DCE006"<<endl;
   cout<<"*****************************************"<<endl;
   cout<<"*\t Programming Assignment 1\t*"<<endl;
```

```
    cout<<"*\t Computer Programming 1\t*"<<endl;
    cout<<"*\t\tAuthor:"<<name<<"\t\t*"<<endl;
    cout<<"*\tDue Date:Thursday,March.3\t*"<<endl;
    cout<<"****************************************";
}
```

Output:



```
C:\c++\1st.exe


22DCE006

***********************************
*   Programming Assignment 1      *
*    Computer  Programming  1     *
*          Author: ??             *
* Due Date: Thursday, Dec.20      *
***********************************
```

Question: Differentiate between \n and endl in two points in below given tabular format:

Sr. No.	\n	Endl
1.	It is a character.	It is a manipulator.
2.	It occupies 1 byte memory as it is a character	It doesn't occupy any memory.

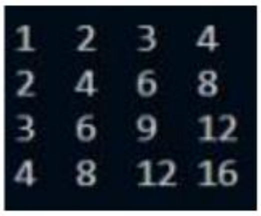
Practical-1.2

Write a program to create the following table by making use of endl and setw manipulator.

1	2	3	4
2	4	6	8
3	6	9	12
4	8	12	16

Expected Output:

Attach the screenshot of output.

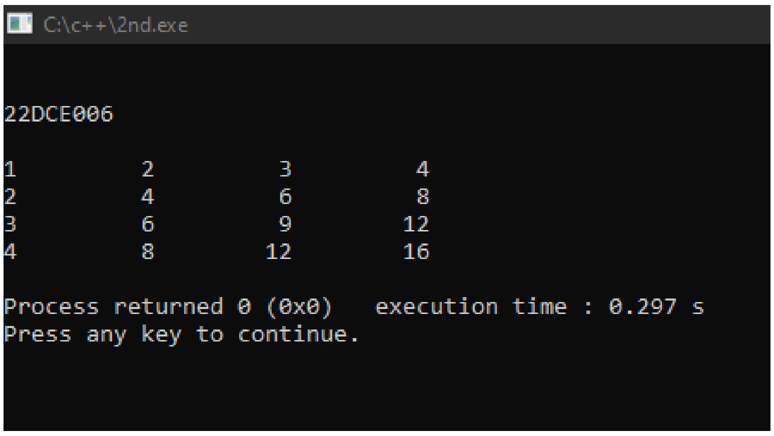


```
1  2  3  4
2  4  6  8
3  6  9  12
4  8  12 16
```

Code:

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
   int i,j;
   cout<<"\n\n22DCE006\n\n";
   for(i=1 ; i<5 ; i++)
   {
      for(j=1 ; j<5 ; j++)
      {
         cout<<i*j<<setw(10);
      }
      cout<<"\n";
   }
   return 0;
}
```

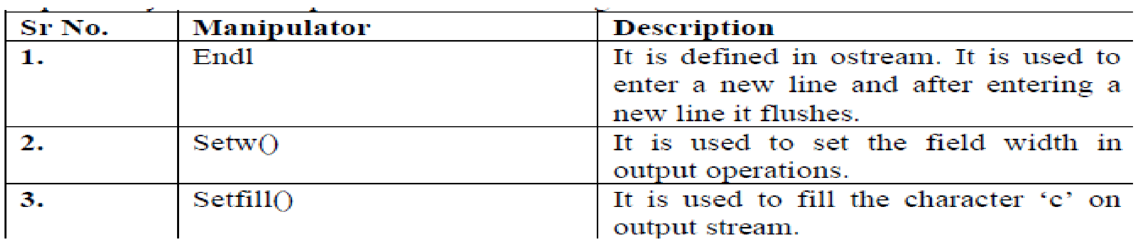
Output:



```
C:\c++\2nd.exe


22DCE006

1         2         3         4
2         4         6         8
3         6         9        12
4         8        12        16

Process returned 0 (0x0)   execution time : 0.297 s
Press any key to continue.
```

Questions:

1. Explain any three manipulators in the below given tabular format.



Sr No.	Manipulator	Description
1.	Endl	It is defined in ostream. It is used to enter a new line and after entering a new line it flushes.
2.	Setw()	It is used to set the field width in output operations.
3.	Setfill()	It is used to fill the character 'c' on output stream.

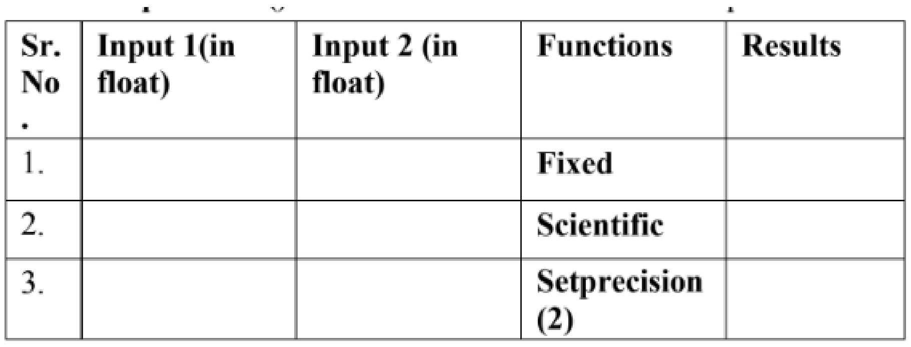
Practical-1.3

Write a C++ program to add two floating numbers using pointer. Theresult should contain only two digits after the decimal.

Note: Use fixed, scientific and setprecision() manipulators for controlling the precision of floating point numbers.

Expected Output:

Fill the following table based on the outcome you get by executing the functions in given sequence- fixed, scientific and setprecision(). Also attach the screenshot of output.

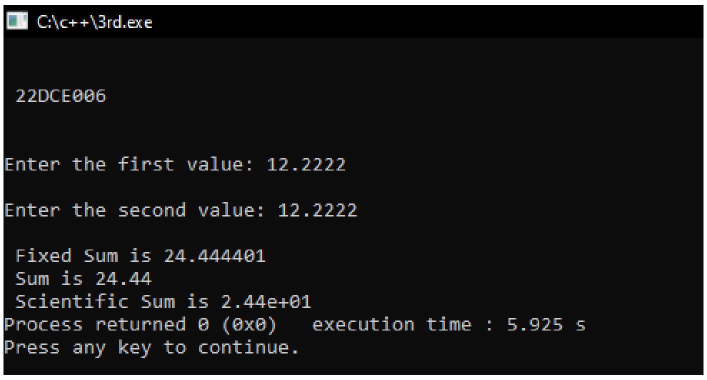


Sr. No.	Input 1(in float)	Input 2 (in float)	Functions	Results
1.			Fixed	
2.			Scientific	
3.			Setprecision (2)	

Code:

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
        float a,b,sum;
        float *x,*y;
        cout<<"\n\n 22DCE006\n\n";
        cout<<"\nEnter the first value: ";
        cin>>a;
        cout<<"\nEnter the second value: ";
        cin>>b;

        x=&a;
        y=&b;
        sum= *x + *y;

        cout<<"\n Fixed Sum is "<<fixed<<sum ;
        cout<<"\n Sum is "<<setprecision(2)<<sum ;
        cout<<scientific<<"\n Scientific Sum is "<<sum ;
        return 0;

}
```

Output:



```
C:\c++\3rd.exe


 22DCE006


Enter the first value: 12.2222

Enter the second value: 12.2222

 Fixed Sum is 24.444401
 Sum is 24.44
 Scientific Sum is 2.44e+01
Process returned 0 (0x0)   execution time : 5.925 s
Press any key to continue.
```

Questions:

1. Which ios class function will be responsible for setting the number of decimal places?

1. Explain any three manipulators in the below given tabular format.

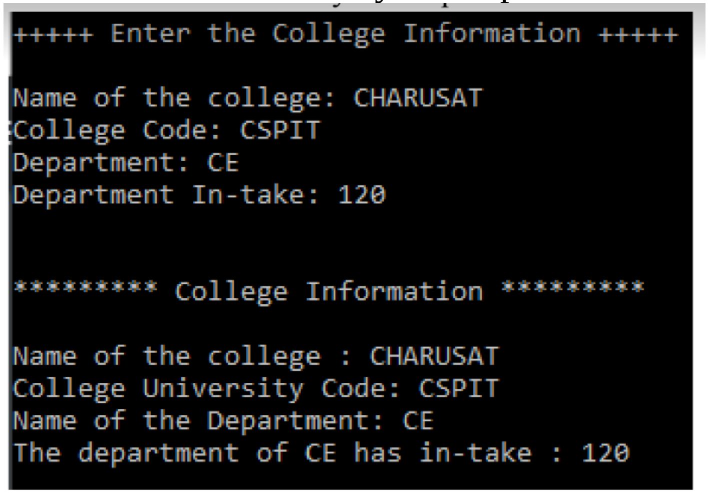
Sr No.	Manipulator	Description
1.	Endl	It is defined in ostream. It is used to enter a new line and after entering a new line it flushes.
2.	Setw()	It is used to set the field width in output operations.
3.	Setfill()	It is used to fill the character 'c' on output stream.

Practical Set- 2

Practical-2.1

Write a C++ Program to declare the struct named College_Details, by taking following data members:
char college_name[10]; (eg. CHARUSAT)
char college_code[10]; (eg. CSPIT/DEPSTAR)
char deparment[5]; (eg. CE/CS/IT)
int intake; (eg. 120)
Collect the data from keyboard and display the same in appropriate view.

Expected output:
You may mention your own college code, department and intake.
Attach the screenshot of your output.

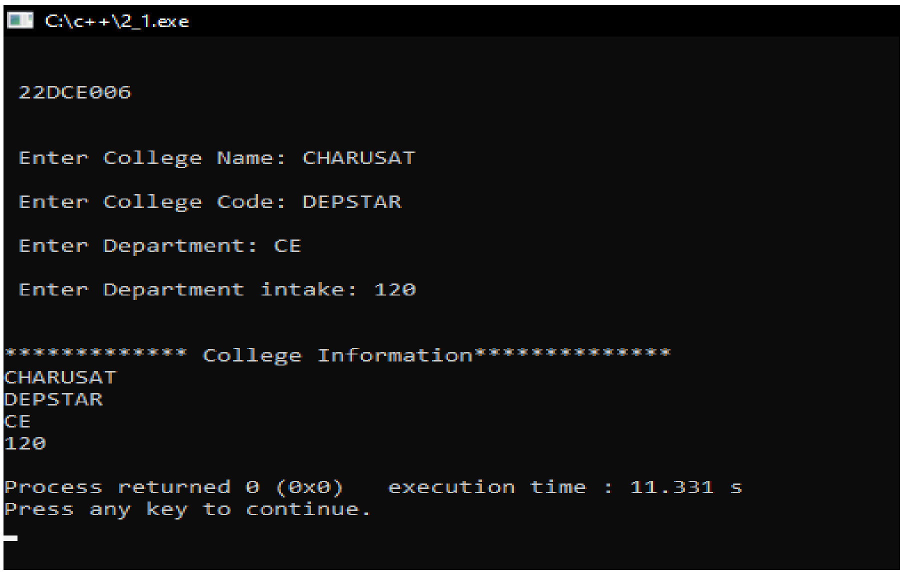


```
+++++ Enter the College Information +++++

Name of the college: CHARUSAT
College Code: CSPIT
Department: CE
Department In-take: 120


********* College Information *********

Name of the college : CHARUSAT
College University Code: CSPIT
Name of the Department: CE
The department of CE has in-take : 120
```

Code:
#include<iostream>
using namespace std;
struct college_details
{

```
    char college_name[10];
    char college_code[10];
    char department[5];
    int intake;
};
int main()
{
    cout<<"\n\n 22DCE006 \n\n ";
    struct college_details c;
    cout<<"\n Enter College Name: ";
    cin>> c.college_name;
    cout<<"\n Enter College Code: ";
    cin>> c.college_code;
    cout<<"\n Enter Department: ";
    cin>> c.department;
    cout<<"\n Enter Department intake: ";
    cin>> c.intake;
    cout<<"\n\n";
    cout<<"************* College Information**************";
    cout<<"\n"<<c.college_name;
    cout<<"\n"<<c.college_code;
    cout<<"\n"<<c.department;
    cout<<"\n"<<c.intake;
    cout<<"\n";
    return 0;
}
```

Output:



```
C:\c++\2_1.exe


 22DCE006


 Enter College Name: CHARUSAT

 Enter College Code: DEPSTAR

 Enter Department: CE

 Enter Department intake: 120


************* College Information**************
CHARUSAT
DEPSTAR
CE
120

Process returned 0 (0x0)   execution time : 11.331 s
Press any key to continue.
```

Questions:
1. Are you able to find the concept of struct that you studied in C Programming, similar to class? If yes, then in which ways?

- **Yes,** declaring and using structure syntax is the same as C Programming. Also, we can use the structure with the same method and it will execute the code the same as the C programming.

Practical -2.2

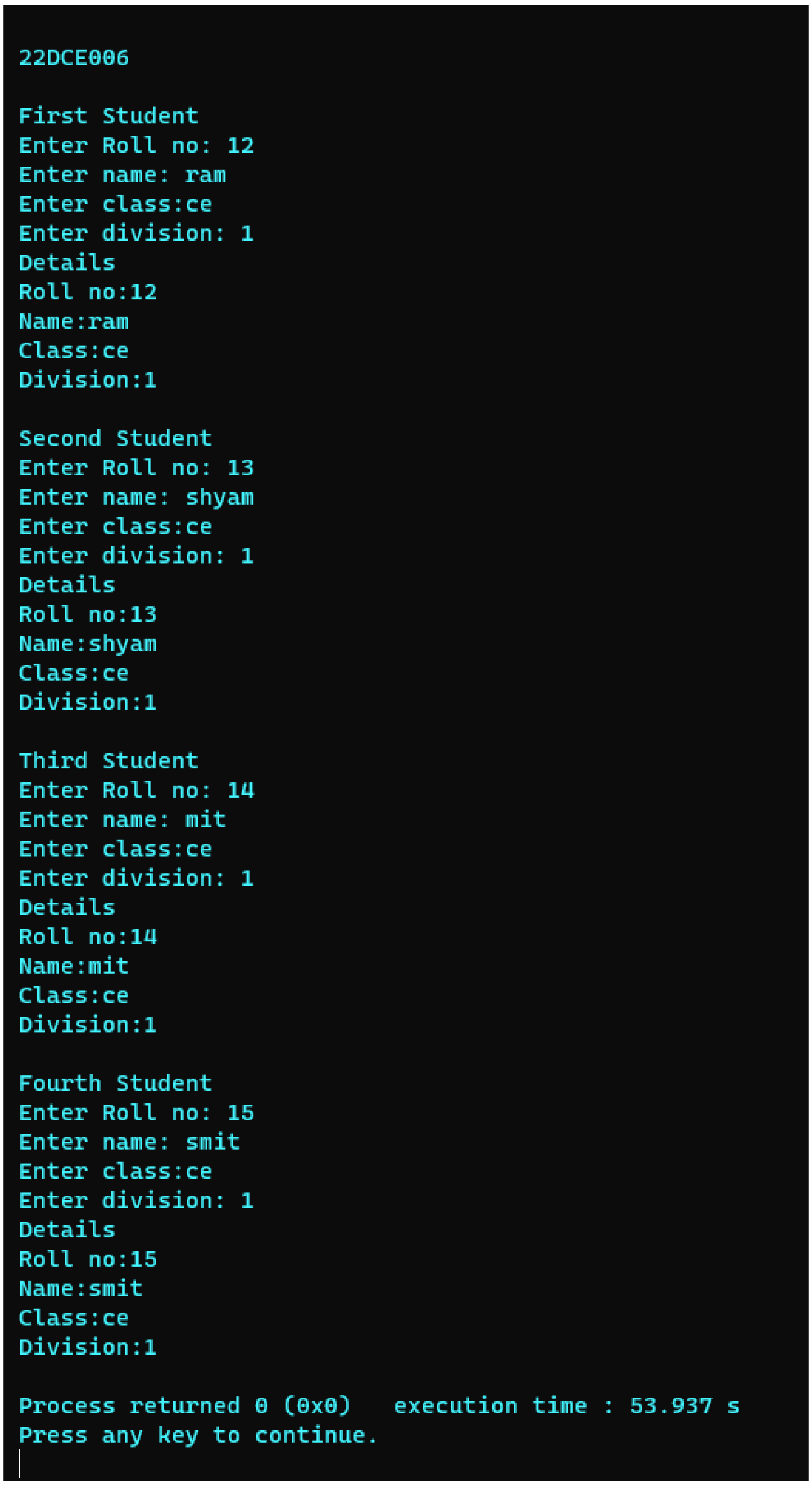
Write a C++ program to collect the details of student like roll_no, name, class and division(A/B) and display the same of 5 students. Declare the following data members in class as public: roll_no, name, class and division(A/B). Make use of two functions read and display as public for collecting information and displaying it respectively.
Note: Create 5 different objects for 5 students.

Expected Output:
Fill the following table to showcase your outcome, also attach the screenshot of output.

Sr. No.	**Name**	**Roll No**	**Class**	**Division(A/B)**
1.				
2.				
3.				
4.				
5.				

Code:
```
#include<iostream>
using namespace std;
class details
{
public:
int roll_no;
char name[10];
char classs[10];
int division;
void read()
{
cout<<"Enter Roll no: ";
cin>>roll_no;
cout<<"Enter name: ";
cin>>name;
cout<<"Enter class:";
cin>>classs;
cout<<"Enter division: ";
cin>>division;
}
void print()
{
cout<<"Details"<<endl;
cout<<"Roll no:"<<roll_no<<endl;
cout<<"Name:"<<name<<endl;
cout<<"Class:"<<classs<<endl;
cout<<"Division:"<<division<<endl;
}
}s1,s2,s3,s4;
int main()
{
cout<<"\n\n22DCE006\n\n";
cout<<"First Student"<<endl;
s1.read();
s1.print();
cout<<endl<<"Second Student"<<endl;
s2.read();
s2.print();
cout<<endl<<"Third Student"<<endl;
s3.read();
s3.print();
cout<<endl<<"Fourth Student"<<endl;
s4.read();
s4.print();
return 0;
}
```

Output:



```
22DCE006

First Student
Enter Roll no: 12
Enter name: ram
Enter class:ce
Enter division: 1
Details
Roll no:12
Name:ram
Class:ce
Division:1

Second Student
Enter Roll no: 13
Enter name: shyam
Enter class:ce
Enter division: 1
Details
Roll no:13
Name:shyam
Class:ce
Division:1

Third Student
Enter Roll no: 14
Enter name: mit
Enter class:ce
Enter division: 1
Details
Roll no:14
Name:mit
Class:ce
Division:1

Fourth Student
Enter Roll no: 15
Enter name: smit
Enter class:ce
Enter division: 1
Details
Roll no:15
Name:smit
Class:ce
Division:1

Process returned 0 (0x0)   execution time : 53.937 s
Press any key to continue.
```

Questions:

1) State the reason for creating object of class.

Ans) When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, you need to create objects.

Practical -2.3

Write a C++ program to swap two numbers without using third variable, using the concept of class and object. Display the values of the variables before and after swapping.

Expected Output:

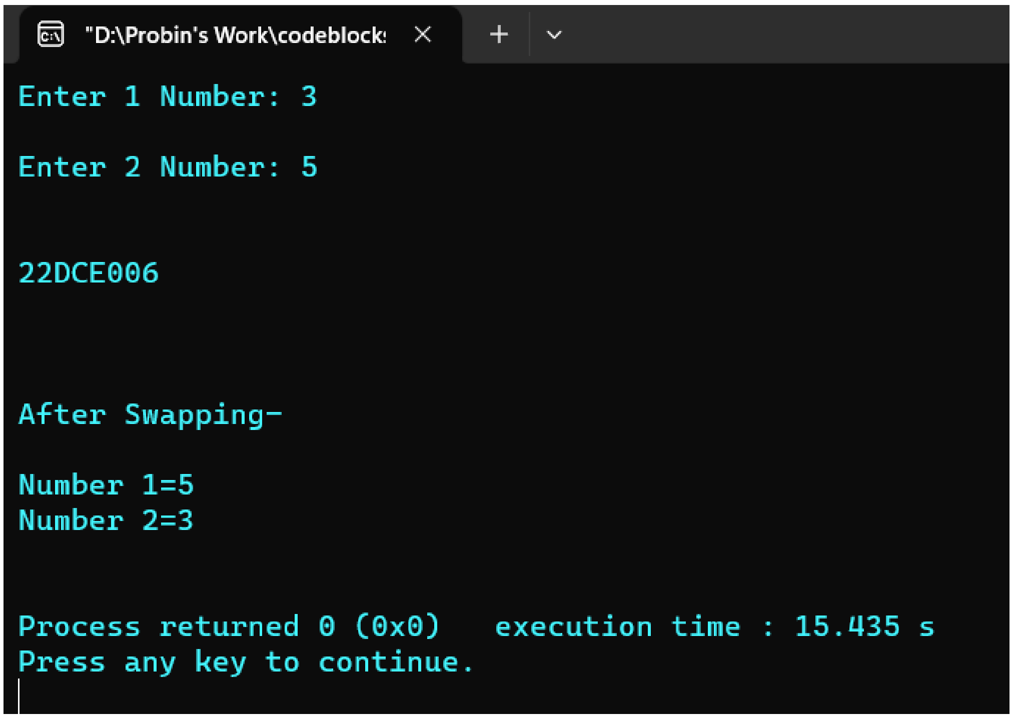
Attach the screenshot of output and fill up the below given table.

Sr. No.	Outcome	Variable_1 value	Variable_2 value
1.	**Before Swapping**		
2.	**After Swapping**		

Code:

```
#include<iostream>
using namespace std;
class swap
{
public:
int a,b;
void number()
{
cout<<"Enter 1 Number: ";
cin>>a;
cout<<endl<<"Enter 2 Number: ";
cin>>b;
}
void display()
{
a=a+b;
b=a-b;
a=a-b;
}
}D;
int main()
{
D.number();
D.display();
cout<<"\n\n22DCE006\n\n";
cout<<endl<<endl<<"After Swapping-"<<endl<<endl;
cout<<"Number 1="<<D.a<<endl;
cout<<"Number 2="<<D.b<<endl<<endl;
return 0;
}
```

Output:



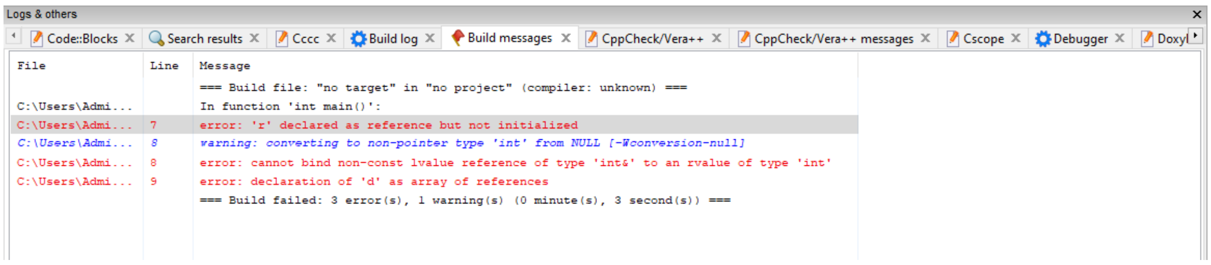
```
Enter 1 Number: 3

Enter 2 Number: 5


22DCE006



After Swapping-

Number 1=5
Number 2=3


Process returned 0 (0x0)   execution time : 15.435 s
Press any key to continue.
```

Practical -3.1

Find error in the following code and give reasons for each error:

```
#include<iostream>
using namespace std;
int main()
{
int no1=10, no2=12;
int & x=no1;
int & r;
int & c = NULL;
int & d[2] = {no1,no2};
cout<<"x = "<< x+20;
cout<<"no1="<< no1+10;
return 0;
}
```

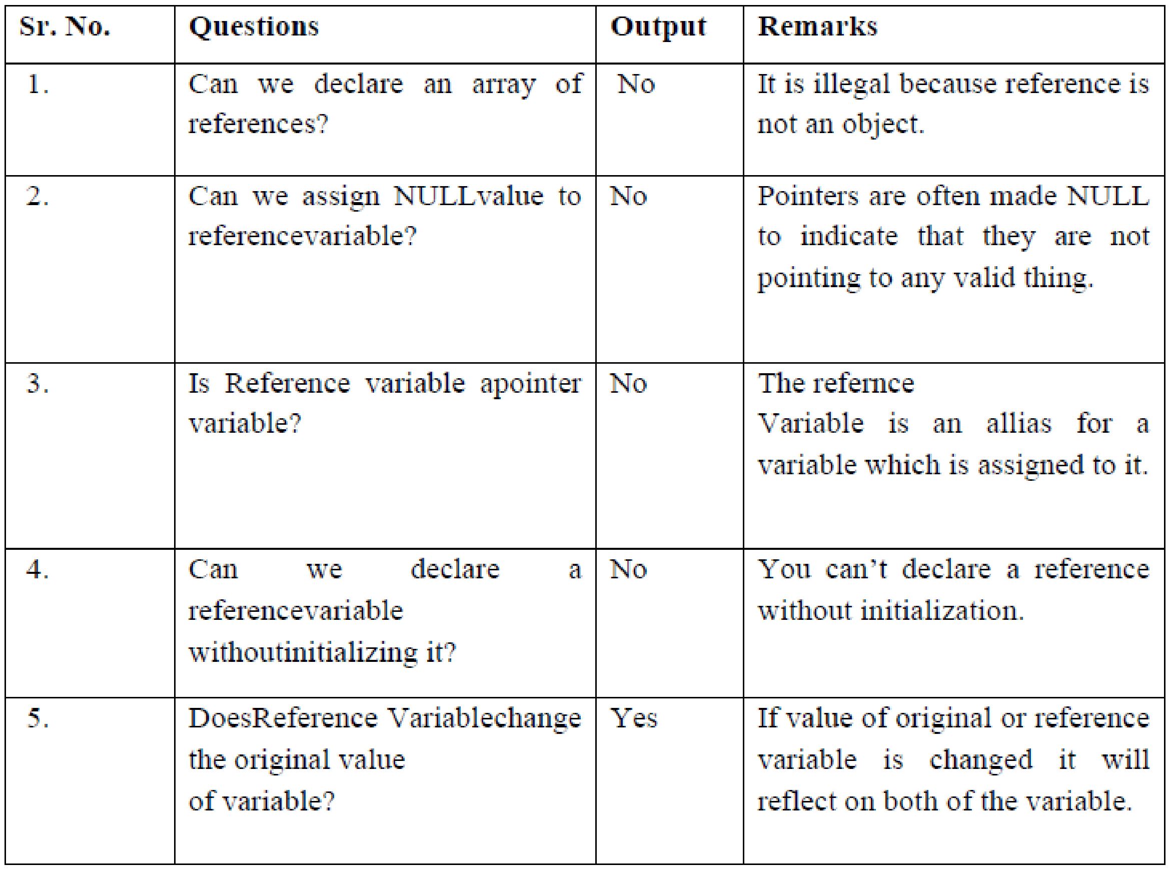
Output:



```
File                Line  Message
                          === Build file: "no target" in "no project" (compiler: unknown) ===
C:\Users\Admi...          In function 'int main()':
C:\Users\Admi...    7     error: 'r' declared as reference but not initialized
C:\Users\Admi...    8     warning: converting to non-pointer type 'int' from NULL [-Wconversion-null]
C:\Users\Admi...    8     error: cannot bind non-const lvalue reference of type 'int&' to an rvalue of type 'int'
C:\Users\Admi...    9     error: declaration of 'd' as array of references
                          === Build failed: 3 error(s), 1 warning(s) (0 minute(s), 3 second(s)) ===
```

Expected Output:

Attach the screenshot of output and fill up the below given table.



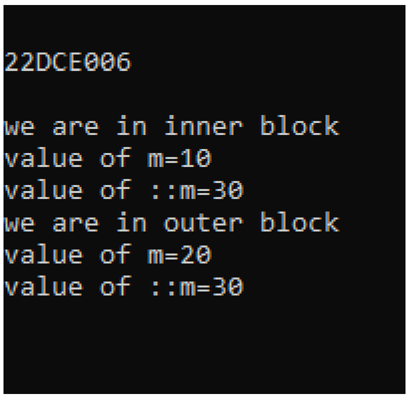
Sr. No.	Questions	Output	Remarks
1.	Can we declare an array of references?	No	It is illegal because reference is not an object.
2.	Can we assign NULLvalue to referencevariable?	No	Pointers are often made NULL to indicate that they are not pointing to any valid thing.
3.	Is Reference variable apointer variable?	No	The refernce Variable is an allias for a variable which is assigned to it.
4.	Can we declare a referencevariable withoutinitializing it?	No	You can't declare a reference without initialization.
5.	DoesReference Variablechange the original value of variable?	Yes	If value of original or reference variable is changed it will reflect on both of the variable.

Practical -3.2

Find output of the following code:

```
#include<iostream.h>
#include<conio.h>
int m=30;
int main()
{
int m=20;
{
int m=10;
cout<<”we are in inner block”<<endl;
cout<<”value of m=”<<m<<”\n”;
cout<<”value of ::m=”<<::m<<”\n”;
}
cout<<"we are in outer block"<<endl;
cout<<"value of m="<<m<<"\n";
cout<<"value of ::m="<<::m<<"\n";
getch();
return 0;
}
```

Attach the screenshot of output.



```
22DCE006

we are in inner block
value of m=10
value of ::m=30
we are in outer block
value of m=20
value of ::m=30
```

Questions:

1. Explain how scope Resolution operator is used to access global

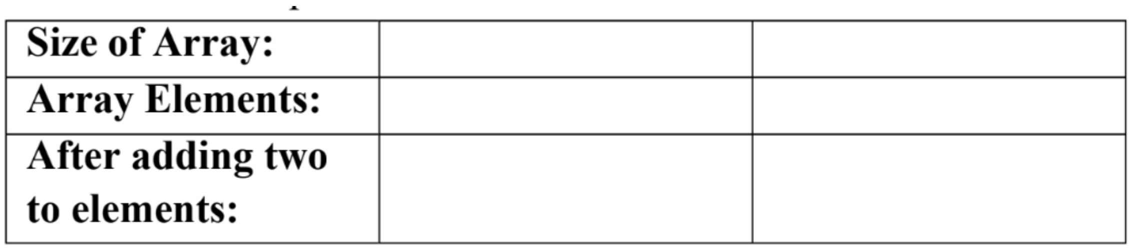
version of a variable.

Ans) The scope resolution operator (::) is used for several reasons. For example: If the global variable name is same as local variable name, the scope resolution operator will be used to call the global variable. It is also used to define a function outside the class and used to access the static variables of class.

Practical - 3.3

Write a program to enter a size of array. Create an array of size given by user using "new" Dynamic memory management operator (free store operator). Enter the data to store in array and display the data after adding 2 to each element in the array. Delete the array by using "delete" memory management operator.

Fill the following table to showcase your outcome, also attach the screenshot of output.

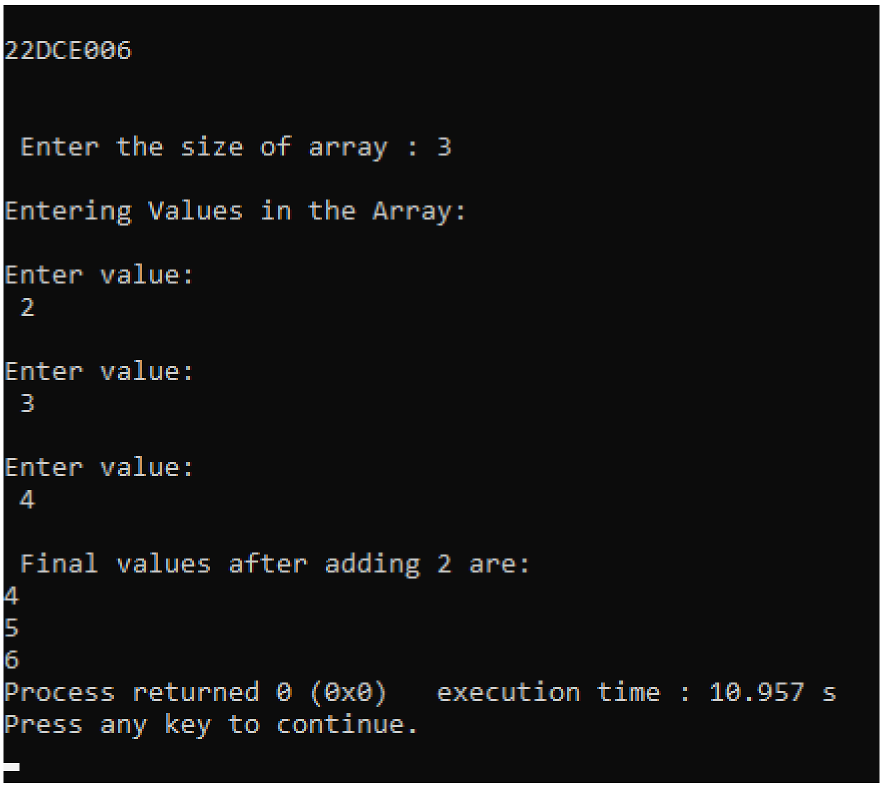


Size of Array:		
Array Elements:		
After adding two to elements:		

Code:

```
#include<iostream>
using namespace std;
int main()
{
   int n,i;
   cout<<"\n\n22DCE006\n\n";
   cout<<"\n Enter the size of array : ";
   cin>>n;
   int *ptr=new int[n];
   cout<<"\nEntering Values in the Array:\n ";
   for(i=0 ; i<n ; i++)
   {
      cout<<"\nEnter value:\n ";
      cin>>*(ptr+i);
   }
   cout<<"\n Final values after adding 2 are:" ;
   for(i=0 ; i<n ; i++)
   {
      cout<<"\n";
      cout<<*(ptr+i)+2;
   }
   delete ptr;
   return 0;
}
```

Output:



```
22DCE006


 Enter the size of array : 3

Entering Values in the Array:

Enter value:
 2

Enter value:
 3

Enter value:
 4

 Final values after adding 2 are:
4
5
6
Process returned 0 (0x0)   execution time : 10.957 s
Press any key to continue.
```

Question :
1. Where the new operator does allocate memory in system?
Ans)New operator denotes the request of memory allocation on the heap , new operator firstly initializes the memory and then returns the address of the newly allocated and initialized memory to the pointer variable. The new operator allocates a memory if sufficient memory is available in the system.

2. State two points on delete operator.
Ans) Delete can be used by either using Delete operator or Delete[] operator.
Delete operator deallocates memory from heap .

Practical Set- 4

Practical-4.1

Define three functions named divide (). First function takes numerator and denominator as an input argument and checks it is divisible or not, second function takes one integer numbers as input argument and checks whether the number is prime or not and Third function takes 3 float number as argument and finds out average of the numbers.

Note: Use concept of Function Overloading / static binding.

Expected Output:

Fill the following table to showcase your outcome, also attach the

screenshot of output.

Display	Input	output
Input two numbers to checkif it is divisible or not	Number 1=32 Number 2=2	Divisible
Input a number to check if it is prime or not	Number =15	Not Prime

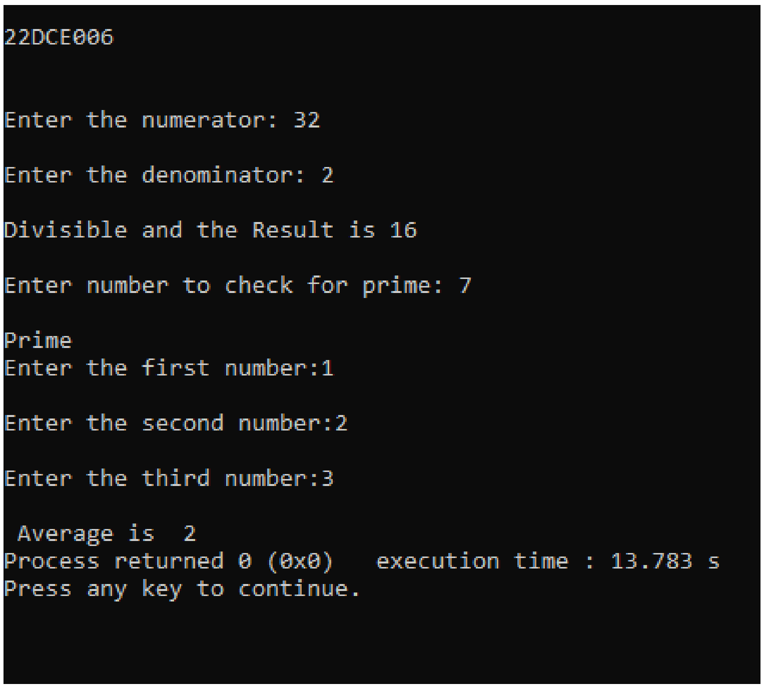
Enter the 3 float number to get average of them	Fnum1=1
	Fnum2=2
	Fnum3=3
		Average=2

Code:

```
#include<iostream>
using namespace std;
int divide(int a , int b)
{
   cout<<"\nEnter the numerator: ";
   cin>>a;
   cout<<"\nEnter the denominator: ";
   cin>>b;
   if(b=0 || b<0 )
   {
      cout<<"\nThe number is not divisible\n ";
   }
   else if(a%b==0)
   {
      cout<<"\nDivisible and the Result is "<<(a/b);
      cout<<"\n";
   }
   else
   {
      cout<<"\nNot Exactly divisible";
   }
   return 0;
}
int divide(int m)
{

   int flag=0;
   cout<<"\nEnter number to check for prime: ";
   cin>>m;
   for(int i=2 ; i<m/2 ; i++)
   {         if(m%i==0)
             flag=1;          }
   if(flag=1)
   { cout<<"\nNot Prime";}
   else
   {cout<<"\nPrime";}
}
int divide(float c, float d , float e )
{
   float avg;
   cout<<"\nEnter the first number:";
   cin>>c;
   cout<<"\nEnter the second number:";
   cin>>d;
   cout<<"\nEnter the third number:";
   cin>>e;
   avg=(c+d+e)/3;
   cout<<"\n Average is  "<<avg;
}
```

```
int main()
{
   cout<<"\n22DCE006\n\n";
   int x,y,finalans1,g,finalans2;
   float h,j,k,finalans3;
   finalans1=divide(x,y);
   finalans1=divide(g);
   finalans1=divide(h,j,k);
}
```

Output:



```
22DCE006


Enter the numerator: 32

Enter the denominator: 2

Divisible and the Result is 16

Enter number to check for prime: 7

Prime
Enter the first number:1

Enter the second number:2

Enter the third number:3

 Average is  2
Process returned 0 (0x0)   execution time : 13.783 s
Press any key to continue.
```

Questions:

1. State the benefits of using function overloading.

Ans) It allows users to have more than one function having the same name but different properties and operations. Overloaded functions enable users to supply different syntax for a function, which depends on the functions.

Conclusion: This practical includes the concept of function overloading, in which when we call the function with the same name, it will call the function of the respective number of arguments that are passed.

Practical-4.2

Write a function called tonLarge () that takes two integer arguments **Call by Reference** and then sets the larger of the two numbers to 100 using **Return by Reference**. Write a main () program to exercise this function.

Expected Output:

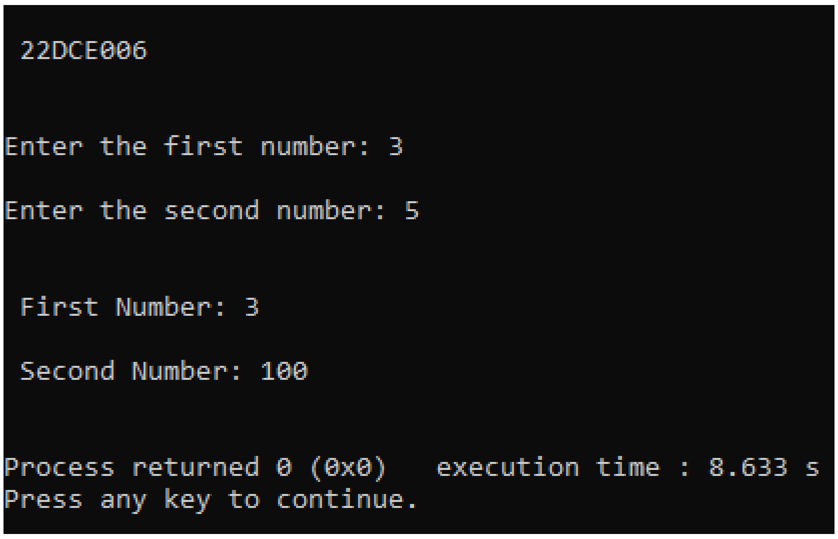
Fill the following table to showcase your outcome, also attach the screenshot of output.

Display	**Input**	**Larger number**	**output**
Enter the two number	Number 1=3	Number1/Number2	Number 1=3
	Number 2=5	Number 2	Number 2=100

Code:

```
#include<iostream>
using namespace std;
int &tonlarge(int &x , int &y)
{
   if(x>y)
         return x;
   else
         return y;
}
int main()
{
cout<<"\n\n 22DCE006\n\n";
int a , b, c;
cout<<"\nEnter the first number: ";
cin>>a;
cout<<"\nEnter the second number: ";
cin>>b;
tonlarge(a,b)=100;
cout<<"\n\n First Number: "<<a;
cout<<"\n\n Second Number: "<<b;
cout<<"\n\n";
}
```

Output:



```
 22DCE006


Enter the first number: 3

Enter the second number: 5


 First Number: 3

 Second Number: 100


Process returned 0 (0x0)   execution time : 8.633 s
Press any key to continue.
```

Questions:

1. Explain the difference of call by reference and return by reference, each in two points.

Ans)

Call by reference:

1. The actual memory address of the variable is passed to the function as a parameter.

2. The function can change the value of the original variable directly.

Return by reference:

1. The function returns a reference to a variable rather than a copy of the variable's value.

2. The caller can use the reference to access and change the original variable outside of the function.

Conclusion: When we pass the reference of a variable to a particular function, it will directly affect the original value. By using this method, we can save memory and also reduces time.

Practical-4.3

Write a inline function called power () that takes two arguments: a double value for **Base** and an integer for **Power**, and returns the result as double value. Use **default argument** as 2 for Base, so that if this argument is omitted, the number will be squared. Write a main () function that gets values from the user to test this function**.**

Expected Output:

Fill the following table to showcase your outcome, also attach the

screenshot of output.

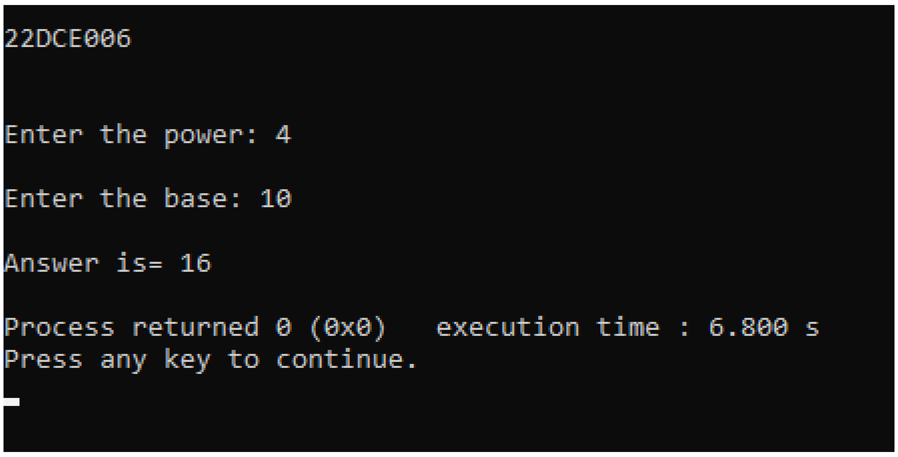
SR NO.	inputs		output
	Enter base	**Enter power**	**Result**
1.	2	4	16
2.	10	4	16

Code:

```
#include<iostream>
#include<iomanip>
using namespace std;
inline int power(int p,int b=2)
{
   int i,result=1;
   for(i=1 ; i<=p ; i++)
   {
      result=result*b;
   }
   cout<<"\nAnswer is= "<<result;
}
int main()
{
   cout<<"\n\n22DCE006\n\n";
   int x,y,ans;
   cout<<"\nEnter the power: ";
   cin>>x;
```

```
    cout<<"\nEnter the base: ";
    cin>>y;
    ans=power(x);
    cout<<"\n";
}
```

Output:



```
22DCE006


Enter the power: 4

Enter the base: 10

Answer is= 16

Process returned 0 (0x0)   execution time : 6.800 s
Press any key to continue.
```

Questions:

Explain the situations where inline function cannot work?

- If a function contains a loop. (for, while, do-while)
- If a function contains static variables.
- If a function is recursive.
- If a function return type is other than void, and the return statement doesn't exist in function body.
- If a function contains switch or goto statement.

Conclusion:

We learned the concept of the default argument. It helps the program to get the default data when the actual argument is ignored in that particular part of the code.

Practical Set-5

Practical-5.1

Write a **C program defining Structure** Rectangle with data member's width and height. It has get values() member functions to get the data from user and area() member functions to print the area of the rectangle. Also create a **C++ Class** for the above program. Define the data members and both functions inside the class. Get **the area of the rectangle** as an output.

Expected Output:

Fill the following table to showcase your outcome, also attach the screenshot of output.

Result using C Structure

Inputs		Outputs
Height	**width**	**Area of rectangle**
10	**6**	**60**

Result using C++ class

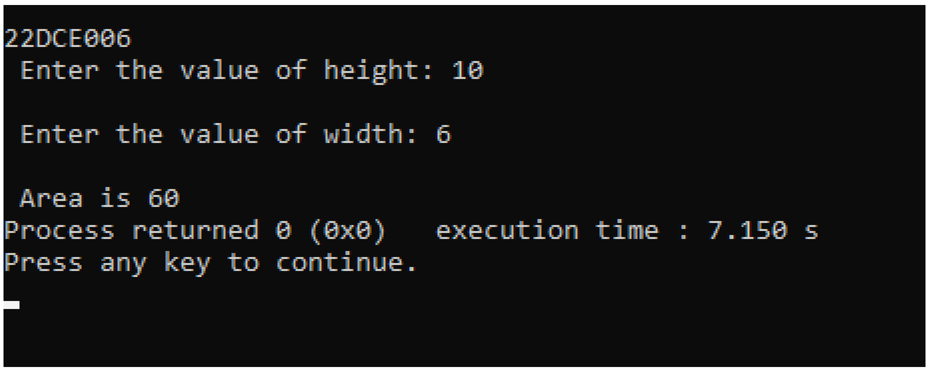
Inputs		Outputs
Height	**Width**	**Area of rectangle**
10	**6**	**60**

Code:

a) Using C language with structure :

```
#include<stdio.h>
struct rectangle
{
   int height;
   int width;
   int area;
}a;
int getvalues()
{
   printf("\n Enter the value of height: ");
   scanf("%d",&a.height);
   printf("\n Enter the value of width: ");
   scanf("%d",&a.width);
}
int area()
{
   a.area=(a.height )*(a.width);
   printf("\n Area is %d",a.area);
}
int main()
{
   printf("\n22DCE006");
   getvalues();
   area();
}
```

Output:



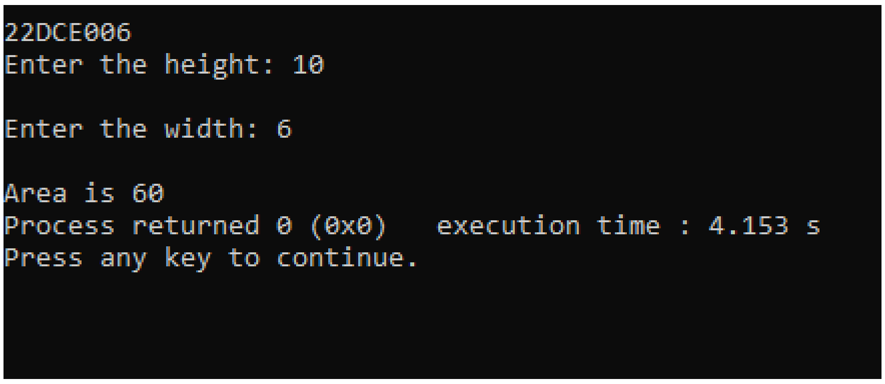
```
22DCE006
 Enter the value of height: 10

 Enter the value of width: 6

 Area is 60
Process returned 0 (0x0)   execution time : 7.150 s
Press any key to continue.
```

b)Using C++ language with class

```
#include<iostream>
using namespace std;
class rectangle
{
   public:
   int height;
   int width;
   int area;
   int getvalues()
   {
      cout<<"\nEnter the height: ";
      cin>>height;
      cout<<"\nEnter the width: ";
      cin>>width;
   }
```

```
    int Area()
    {
       area=height*width;
       cout<<"\nArea is "<<area;
    }
};
int main()
{
    rectangle a;
    cout<<"\n22DCE006";
    a.getvalues();
    a.Area();
    return 0;
}
```

Output:

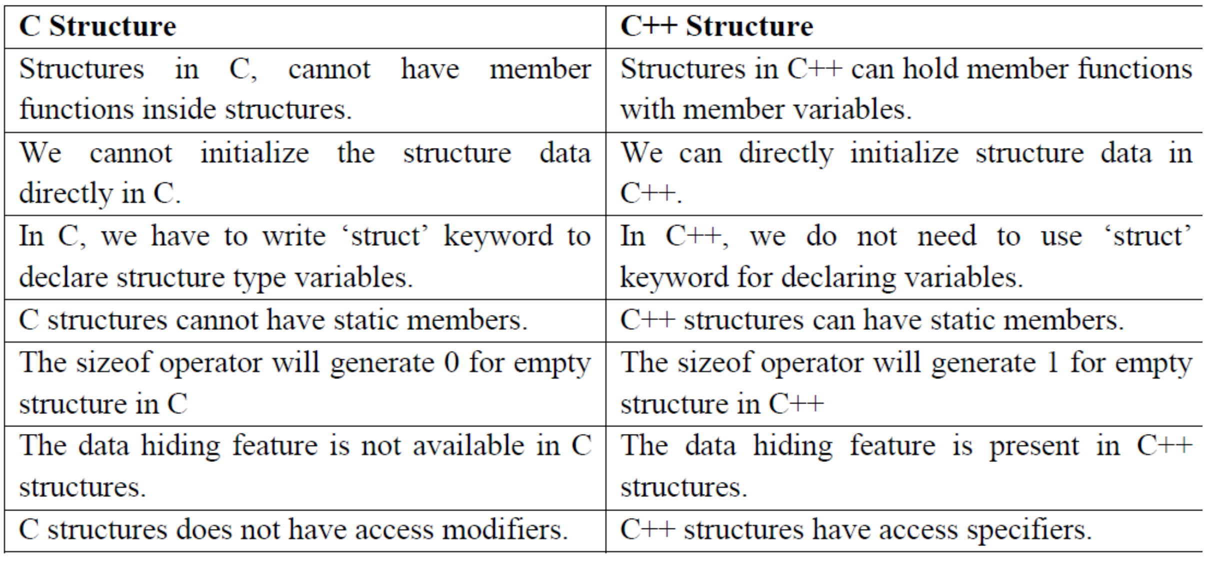


```
22DCE006
Enter the height: 10

Enter the width: 6

Area is 60
Process returned 0 (0x0)   execution time : 4.153 s
Press any key to continue.
```

Questions:

1)Illustrate the difference between C Structure and C++ Class.

Ans-)



C Structure	C++ Structure
Structures in C, cannot have member functions inside structures.	Structures in C++ can hold member functions with member variables.
We cannot initialize the structure data directly in C.	We can directly initialize structure data in C++.
In C, we have to write 'struct' keyword to declare structure type variables.	In C++, we do not need to use 'struct' keyword for declaring variables.
C structures cannot have static members.	C++ structures can have static members.
The sizeof operator will generate 0 for empty structure in C	The sizeof operator will generate 1 for empty structure in C++
The data hiding feature is not available in C structures.	The data hiding feature is present in C++ structures.
C structures does not have access modifiers.	C++ structures have access specifiers.

CONCLUSION:
The difference between a structure and a class is that structure members have public access by default and class members have private access by default. Using the concept of classes is useful in complex programs.

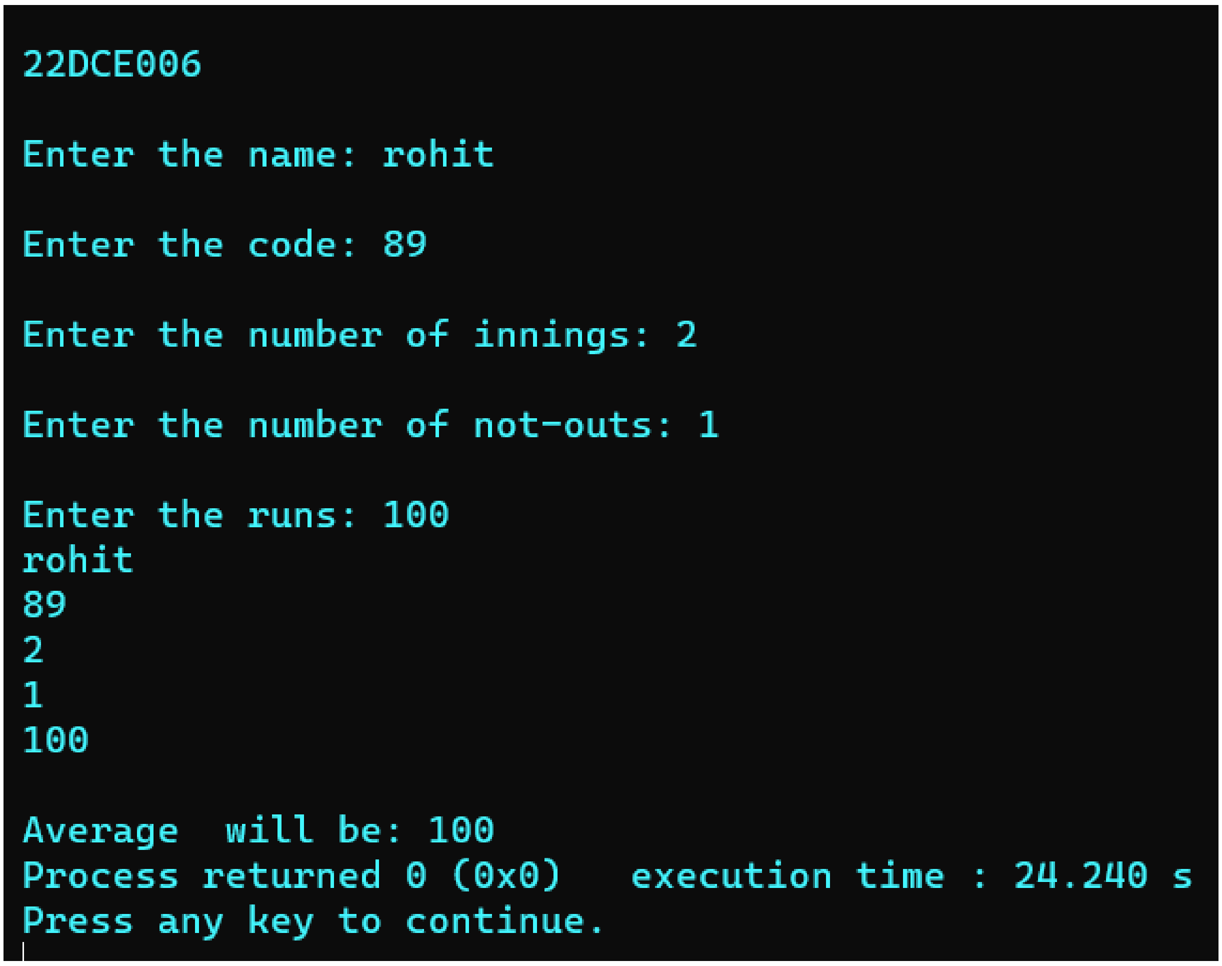
Practical-5.2

Write a C++ program having class **Batsman**. It has private data members: batsman_name, bcode (4 Digit Code Number), innings, not_out, runs, batting average. Innings, not out and runs are in integer and batting_average is in float. Define following **function outside the class using scope resolution operator.**
1) Public member function getdata() to read values of data members.
2) Public member function putdata() to display values of data
members.
3) **Private member function** calcavg() which calculates the batting
average of a batsman. Also make this outside function **inline.**
Hint: batting_average = runs/(innings – not_out)
Expected Output:
Fill the following table to showcase your outcome, also attach the screenshot of output.

Parameters	**Inputs**	**Outputs(Batting Average)**
Name	Rohit	100
Bcode	89	
Total Innings	2	
Enter not_out_timings	1	
Enter total runs	100	

Code:
```
#include<iostream>
using namespace std;
class batsman
{
private:
    char name[20];
    int bcode;
    int innings;
    int not_out;
    int runs;
    float average;
public:
    inline int getdata();
    inline int putdata();
private:
    float calcavg();
};
int batsman:: getdata()
    {
        cout<<"\nEnter the name: ";
        cin>>name;
        cout<<"\nEnter the code: ";
        cin>>bcode;
        cout<<"\nEnter the number of innings: ";
        cin>>innings;
        cout<<"\nEnter the number of not-outs: ";
        cin>>not_out;
        cout<<"\nEnter the runs: ";
        cin>>runs;
    }
     int batsman::putdata()
    {
        cout<<name;
        cout<<"\n";
        cout<<bcode;
        cout<<"\n";
        cout<<innings;
        cout<<"\n";
        cout<<not_out;
        cout<<"\n";
        cout<<runs;
        cout<<"\n";
        calcavg();
    }
    float batsman::calcavg()
    {
        average=runs/(innings-not_out);
        cout<<"\nAverage  will be: "<<average;
    }
int main()
{
    class batsman a;
    cout<<"\n22DCE006\n";
    a.getdata();
    a.putdata();
}
```

Output:



```
22DCE006

Enter the name: rohit

Enter the code: 89

Enter the number of innings: 2

Enter the number of not-outs: 1

Enter the runs: 100
rohit
89
2
1
100

Average  will be: 100
Process returned 0 (0x0)   execution time : 24.240 s
Press any key to continue.
```

CONCLUSION:
When we define member functions outside the class we need to use scope resolution operator and to make a function inline we need to use inline keyword in the beginning of he declaration.

Practical-5.3

Define class Currency having two integer data members rupee and paisa. A class has member functions enter() to get the data and show() to print the amount in 22.50 format. Define one member function sum() that adds two objects of the class and stores answer in the third object i.e. c3=c1.sum(c2). The second member function should add two objects of type currency passed as arguments such that it supports c3.add(c1, c2); where c1, c2 and c3 are objects of class Currency. Also Validate your answer if paisa >100. Write a main( ) program to test all the functions.

Use concepts of Object as Function Arguments, function returning object and function overloading.

Expected Output:

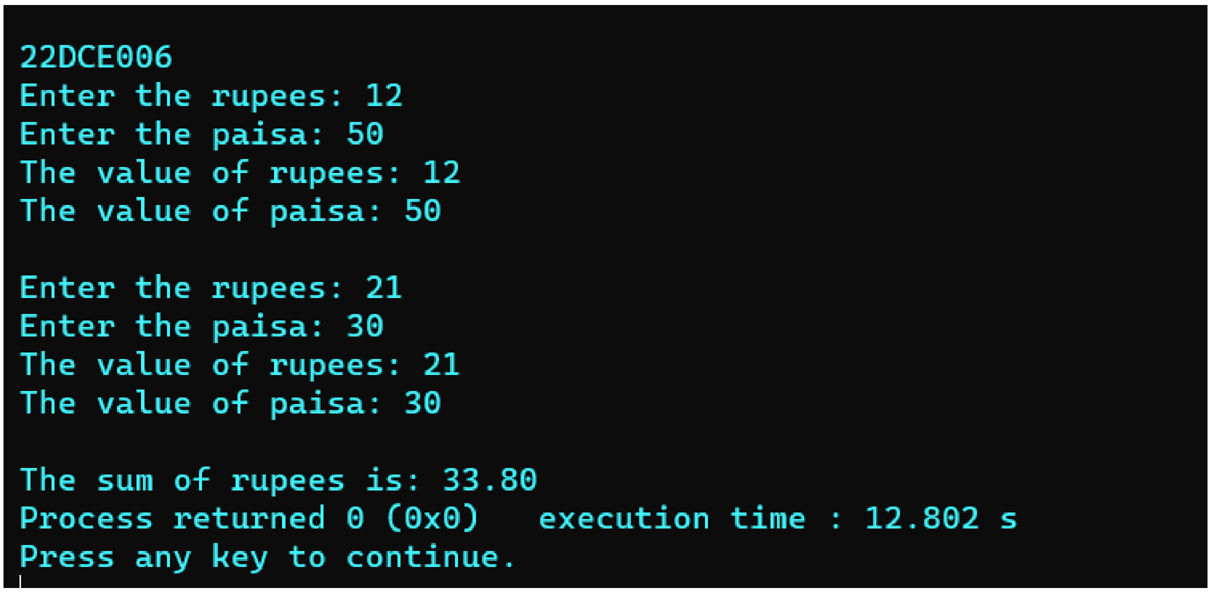
Fill the following table to showcase your outcome, also attach the screenshot of output.

Using sum()		
Rupees	**Paisa**	**Total Amount**
Using add()		
Rupees	**Paisa**	**Total Amount**

Code:

```
#include<iostream>
using namespace std;
class currency
{
public:
    int rupees,paisa;
    void getdata()
    {
        cout<<"Enter the rupees: ";
        cin>>rupees;
        cout<<"Enter the paisa: ";
        cin>>paisa;
    }
    void putdata()
    {
        cout<<"The value of rupees: "<<rupees<<endl;
        cout<<"The value of paisa: "<<paisa<<endl;
    }
    void add(currency c1,currency c2)
    {
        currency c3;
        rupees=c1.rupees+c2.rupees;
        paisa=c1.paisa+c2.paisa;
        c3.paisa=(c3.paisa)/100;
        rupees=rupees+((paisa)/100);
        paisa=(paisa)%100;
        cout<<"The sum of rupees is: "<<rupees<<"."<<paisa;
    }
};
int main()
{
    cout<<"\n\n22DCE006\n";
    currency c1,c2,c3;
    c1.getdata();
    c1.putdata();
    cout<<endl;
    c2.getdata();
    c2.putdata();
    cout<<endl;
    c3.add(c1,c2);
}
```

Output:



```
22DCE006
Enter the rupees: 12
Enter the paisa: 50
The value of rupees: 12
The value of paisa: 50

Enter the rupees: 21
Enter the paisa: 30
The value of rupees: 21
The value of paisa: 30

The sum of rupees is: 33.80
Process returned 0 (0x0)   execution time : 12.802 s
Press any key to continue.
```

CONCLUSION: From this program I learnt to pass the objects as function arguments and returning object. Additionally, when we define more than one function with the same name but with the different arguments it is called the concept of function overloading.

Practical-5.4

Define a class Dist with int feet and float inches. Define member function that displays distance in 1’-2.5” format. Also define member function scale ( ) function that takes object by reference and scale factor in float as an input argument. The function will scale the distance accordingly.
For example, 20’-5.5” and Scale Factor is 0.5 then answer is 10’-2.75”

Expected Output:

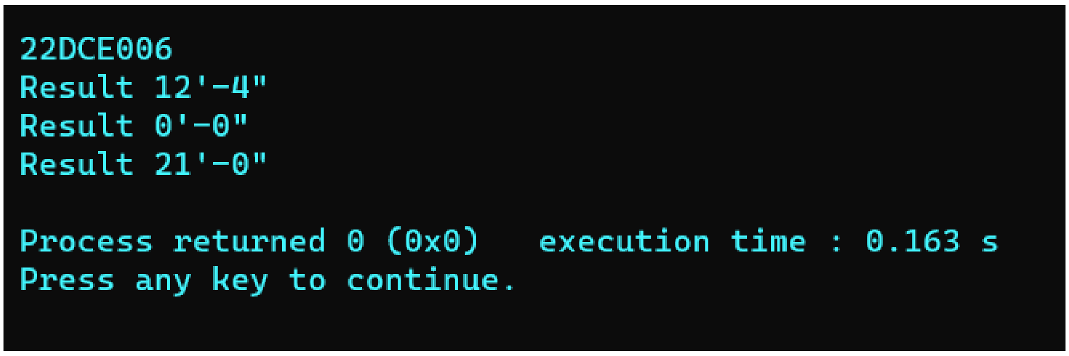
Fill the following table to showcase your outcome as per inputs given, also attach the screenshot of output.

Feet	Inches	Scaling Factor	Output (Distance)
6	2	2	
3	5	0	
7	0	3	

Code:

```
#include<iostream>
using namespace std;
class Dist
{
    int feet;
    float inches,sf;
    public:
    void getdata(int ft,float in)
    {
        feet=ft;
        inches=in;
    }
    void scale(float s)
    {
        sf=s;
    }
    void calculate(Dist d)
    {
        Dist ans;
        ans.feet=d.feet*d.sf;
        ans.inches=d.inches*d.sf;
        if(ans.inches>12)
        {
            ans.feet++;
            ans.inches=ans.inches-12;
            cout<<"Result "<<ans.feet<<"\'"<<"-"<<ans.inches<<"\""<<endl;
        }
        else
        {
            cout<<"Result "<<ans.feet<<"\'"<<"-"<<ans.inches<<"\""<<endl;
        }
    }
};
int main()
{
    Dist d1,d2,d3;
    cout<<"\n22DCE006\n";
    d1.getdata(6,2);
    d1.scale(2);
    d1.calculate(d1);
    d2.getdata(3,5);
    d2.scale(0);
    d2.calculate(d2);
    d3.getdata(7,0);
    d3.scale(3);
    d3.calculate(d3);
}
```

Output:



```
22DCE006
Result 12'-4"
Result 0'-0"
Result 21'-0"

Process returned 0 (0x0)   execution time : 0.163 s
Press any key to continue.
```

CONCLUSION: When we define member functions by passing reference, it will directly affect the original value of a variable. Therefore, the function works with the original value and hence it becomes easy to for programmer to read and execute.

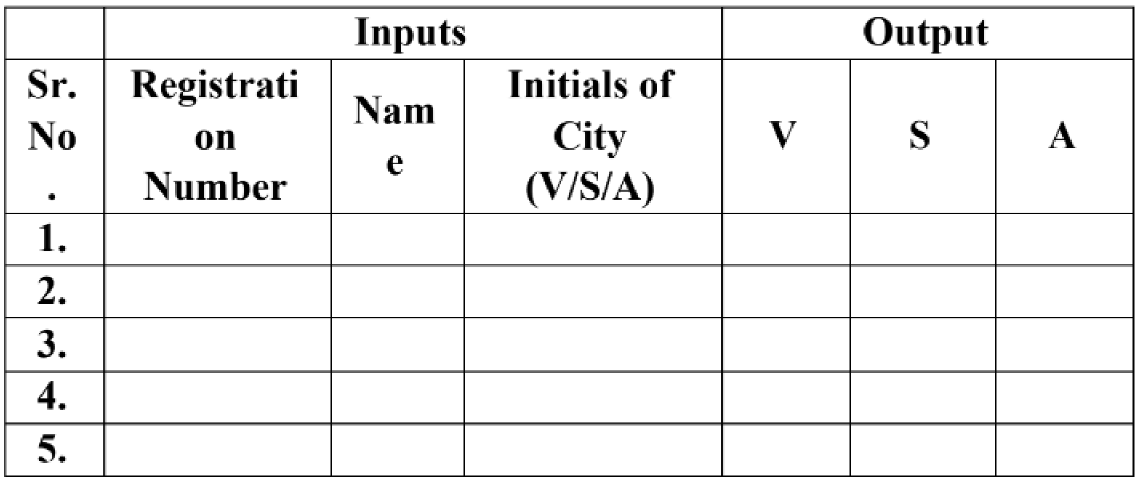
Practical-5.5

Create a Class Gate for students appearing in Gate (Graduate Aptitude test for Engineering) exam. There are three examination center Vadodara, Surat, and Ahmedabad where Gate exams are conducted. A class has data members: Registration number, Name of student, Examination center. Class also Contains static data member ECV_Cnt, ECS_Cnt and ECA_Cnt which counts the number of students in Vadodara, Surat and Ahmedabad exam center respectively. Class Contains two Member function getdata () which gets all information of students and counts total students in each exam center and pudata () which prints all information about the students. Class also contains one static member function getcount () which displays the total number of students in each examination center. Write a program for 5 students and display the total number of students in each examination center.

Use static data member, static member function and Array of Objects.

Expected Output:

Fill the following table to showcase your outcome, also attach the screenshot of output.



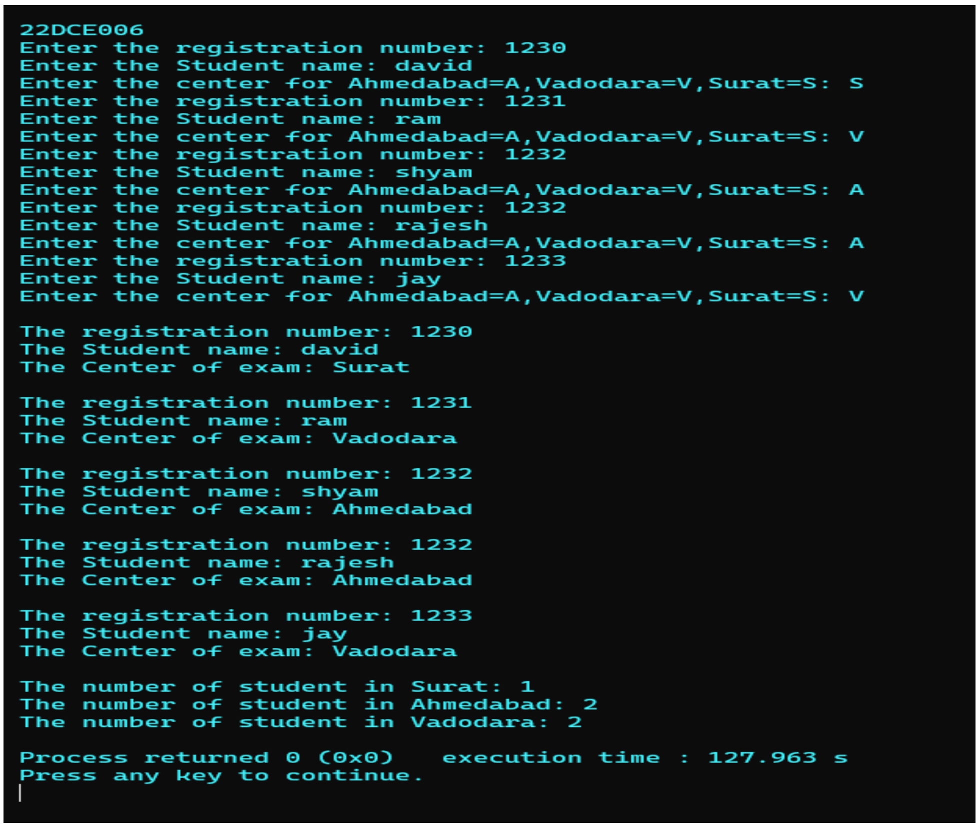
	Inputs			Output		
Sr. No.	Registration Number	Name	Initials of City (V/S/A)	V	S	A
1.						
2.						
3.						
4.						
5.						

Code:

```
#include<iostream>

using namespace std;

class Gate

{

public:

   int regi_nu;

   char name[20];
   char center;
   static int ECS_cnt;
   static int ECA_cnt;
   static int ECV_cnt;
   void getdata()
   {
       cout<<"Enter the registration number: ";
       cin>>regi_nu;
       cout<<"Enter the Student name: ";
       cin>>name;
       cout<<"Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: ";
       cin>>center;
       if(center=='S' || center=='s')
       {
       ECS_cnt++;
       }
       else if(center=='A' || center=='a')
       {
       ECA_cnt++;
       }
       else
       {
       ECV_cnt++;
       }
   }
   void putdata()
   {
        cout<<endl<<"The registration number: "<<regi_nu<<endl;
        cout<<"The Student name: "<<name<<endl;
        cout<<"The Center of exam: ";
        if(center=='S' || center=='s')
        {
            cout<<"Surat"<<endl;
        }
        else if(center=='A' || center=='a')
        {
            cout<<"Ahmedabad"<<endl;
        }
        else
        {
            cout<<"Vadodara"<<endl;
        }
    }
    static void getcount()
    {
        cout<<endl<<"The number of student in Surat: "<<ECS_cnt<<endl;
        cout<<"The number of student in Ahmedabad: "<<ECA_cnt<<endl;
        cout<<"The number of student in Vadodara: "<<ECV_cnt<<endl;
    }
};
    int Gate :: ECS_cnt=0;
    int Gate :: ECA_cnt=0;
    int Gate :: ECV_cnt=0;
int main()
```

```
{
   Gate n[5];
   cout<<"\n22DCE006\n";
   int i;
   for(i=0;i<5;i++)
   {
   n[i].getdata();
   }
   for(i=0;i<5;i++)
   { n[i].putdata(); }
Gate :: getcount();
}
```

Output:



```
22DCE006
Enter the registration number: 1230
Enter the Student name: david
Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: S
Enter the registration number: 1231
Enter the Student name: ram
Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: V
Enter the registration number: 1232
Enter the Student name: shyam
Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: A
Enter the registration number: 1232
Enter the Student name: rajesh
Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: A
Enter the registration number: 1233
Enter the Student name: jay
Enter the center for Ahmedabad=A,Vadodara=V,Surat=S: V

The registration number: 1230
The Student name: david
The Center of exam: Surat

The registration number: 1231
The Student name: ram
The Center of exam: Vadodara

The registration number: 1232
The Student name: shyam
The Center of exam: Ahmedabad

The registration number: 1232
The Student name: rajesh
The Center of exam: Ahmedabad

The registration number: 1233
The Student name: jay
The Center of exam: Vadodara

The number of student in Surat: 1
The number of student in Ahmedabad: 2
The number of student in Vadodara: 2

Process returned 0 (0x0)   execution time : 127.963 s
Press any key to continue.
```

CONCLUSION: We learnt the concept of static data member, static member function and array of objects and successfully using them in the above program.

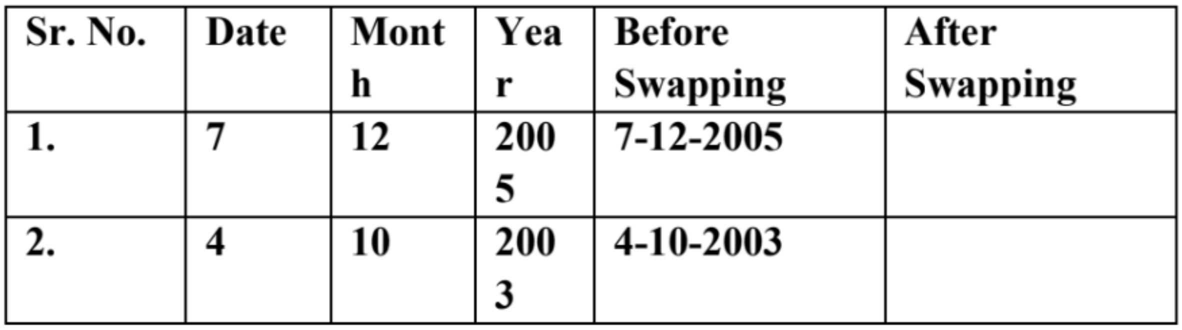
Practical-5.6

Create a Class Date having data members: int dd, mm, yyyy. Class has one member function to input the dates and another member function which prints the dates. Write a main() function which takes two dates as
input. Write a friend function swapdates() which takes two objects by reference of type Date and swaps both the dates.

Use the concept of Friend function which takes objects by reference

Expected Output:

Fill the following table to showcase your outcome as per the given
inputs, also attach the screenshot of output.



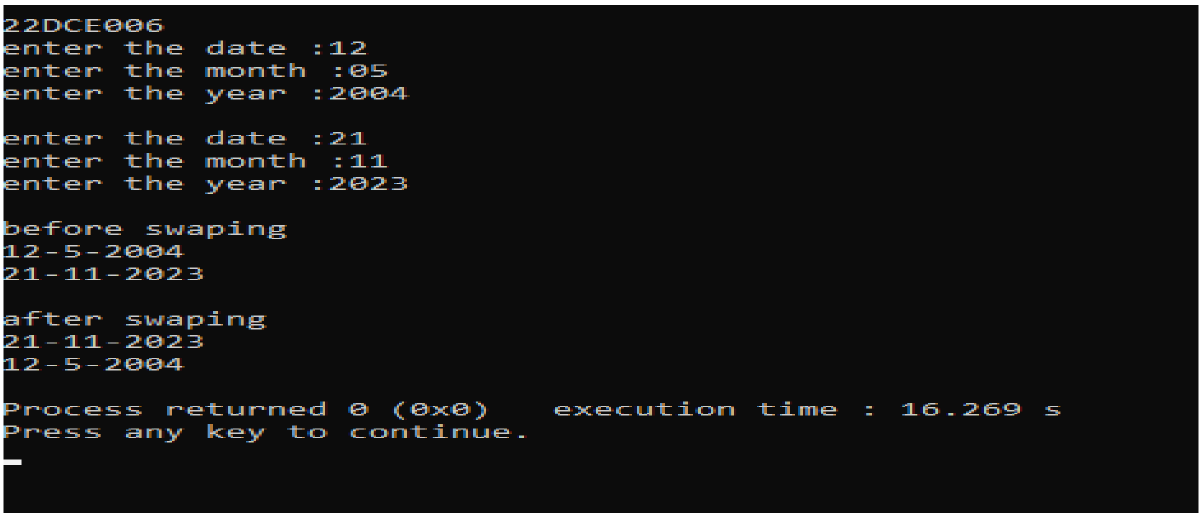
Sr. No.	Date	Month	Year	Before Swapping	After Swapping
1.	7	12	2005	7-12-2005	
2.	4	10	2003	4-10-2003	

Code:

```
#include<iostream>
using namespace std;
class date
{
   int dd,mm,yyyy;
   friend void swapdate(date &d1,date &d2);
   public:
   void get()
   {
      cout<<"enter the date :";
      cin>>dd;
      cout<<"enter the month :";
      cin>>mm;
      cout<<"enter the year :";
      cin>>yyyy;
   }
   void put()
   {
      cout<<dd<<"-"<<mm<<"-"<<yyyy<<endl;
   }
};
```

```
void swapdate(date &d1,date &d2)
 {
    date d;
    d=d1;
    d1=d2;
    d2=d;
 }
 int main()
 {
    cout<<"\n22DCE006\n";
    date d1,d2;
    d1.get();
    cout<<endl;
    d2.get();
    cout<<endl<<"before swaping"<<endl;
    d1.put();
    d2.put();
    cout<<endl<<"after swaping"<<endl;
    swapdate(d1,d2);
    d1.put();
    d2.put();
 }
```

Output:



```
22DCE006
enter the date :12
enter the month :05
enter the year :2004

enter the date :21
enter the month :11
enter the year :2023

before swaping
12-5-2004
21-11-2023

after swaping
21-11-2023
12-5-2004

Process returned 0 (0x0)   execution time : 16.269 s
Press any key to continue.
```

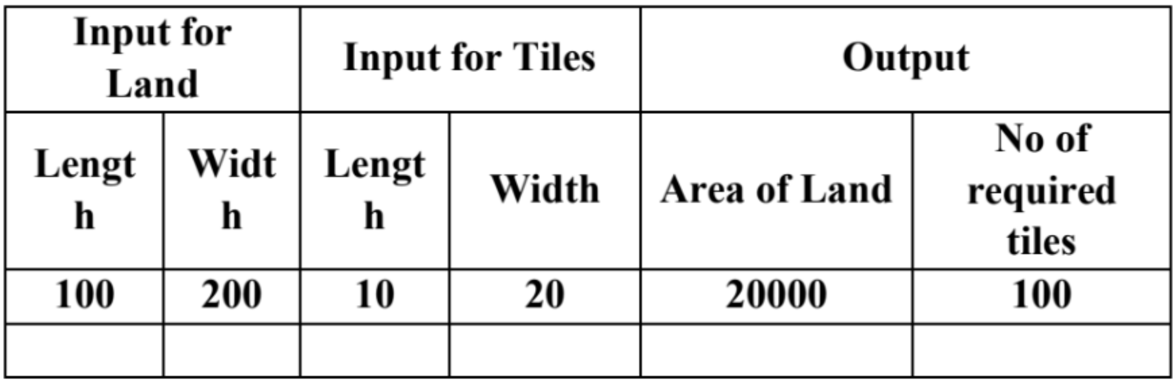
CONCLUSION: I learnt to use quite useful concept of friend function which takes objects by reference.

Practical-5.7

Create a class LAND having data members: length, width, area1. Write member functions to read and display the data of land. Also, calculates the area of the land. Create another class TILES having data members: l, w, area2. Write a member function to get the data of tile. Calculate the area of one tile. Class TILE has a member function named number_of_tiles() which is a friend of class LAND and takes the object of class LAND by reference which calculates the number of tiles which can be put over the land area. Write the main function to test all the functions. **Use the concept of member function of one class can be a friend function of another class.**

Expected Output:

Fill the following table to showcase your outcome, also attach the screenshot of output. Sample Input and Output are stated below. Try one by your self and fill up the table.



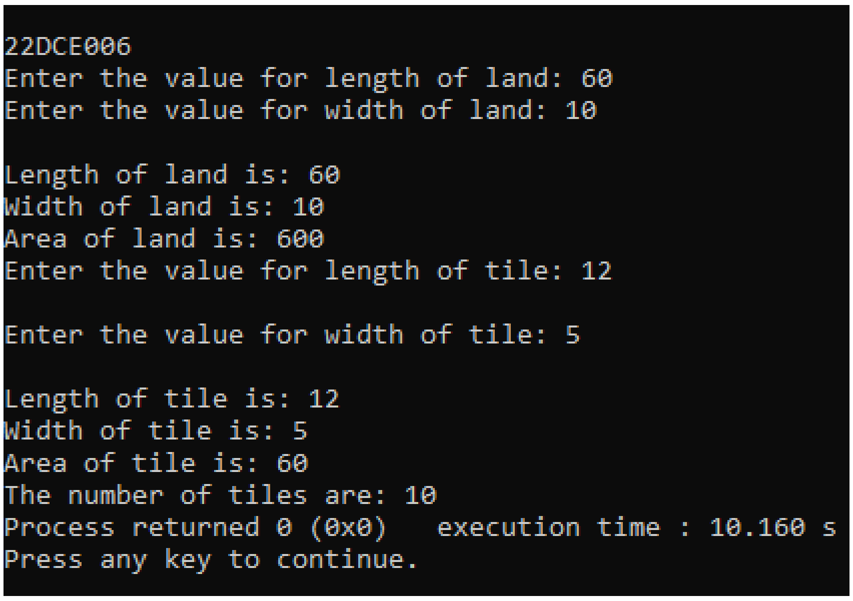
Input for Land		Input for Tiles		Output	
Lengt h	Widt h	Lengt h	Width	Area of Land	No of required tiles
100	200	10	20	20000	100

Code:

```
#include<iostream>
using namespace std;
class land;
class tiles
{
   int tl,tw,a2;
   public:
   void getdata2()
   {
      cout<<"\nEnter the value for length of tile: ";
      cin>>tl;
      cout<<"\nEnter the value for width of tile: ";
      cin>>tw;
```

```
        a2=tl*tw;
    }
    void putdata2()
    {
        cout<<"\nLength of tile is: "<<tl;
        cout<<"\nWidth of tile is: "<<tw;
        cout<<"\nArea of tile is: "<<a2;
    }
    void number_of_tiles(land&);
}y;
class land
{
    int length,width,a1;
    public:
    void getdata1()
    {
        cout<<"Enter the value for length of land: ";
        cin>>length;
        cout<<"Enter the value for width of land: ";
        cin>>width;
        a1=length*width;
    }
    void putdata1()
    {
        cout<<"\nLength of land is: "<<length;
        cout<<"\nWidth of land is: "<<width;
        cout<<"\nArea of land is: "<<a1;
    }
    friend void tiles::number_of_tiles(land &a);
}x;
void tiles::number_of_tiles(land &a)
    {       int ans=a.a1/a2;
        cout<<"\nThe number of tiles are: "<<ans;
    }
int main()
{
    cout<<"\n22DCE006\n";
    x.getdata1();
    x.putdata1();
    y.getdata2();
    y.putdata2();
    y.number_of_tiles(x);
}
```

Output:



```
22DCE006
Enter the value for length of land: 60
Enter the value for width of land: 10

Length of land is: 60
Width of land is: 10
Area of land is: 600
Enter the value for length of tile: 12

Enter the value for width of tile: 5

Length of tile is: 12
Width of tile is: 5
Area of tile is: 60
The number of tiles are: 10
Process returned 0 (0x0)   execution time : 10.160 s
Press any key to continue.
```

CONCLUSION: We learnt the concept of member function of one class can be a friend function of another class from this program.

Practical-5.8

Create a class Child having data members: name of the child and gender and a member function to get and print child data. Create another class Parent which is a friend class of child class. Class Parent have member function ReadChildData() which takes child’s object by reference as input argument and Reads the childs data and DisplayChildData() which takes childs object as argument and displays childs data. Use the concepts of Friend Class.

Expected Output:

Fill the following table to showcase your outcome, also attach the screenshot of output. Sample Input and Output are stated below. Try one by your self and fill up the table.

Input		Output	
Name	**Gender**	**Name**	**Gender**
Aarya	**Dutta**	**Aarya**	**Dutta**

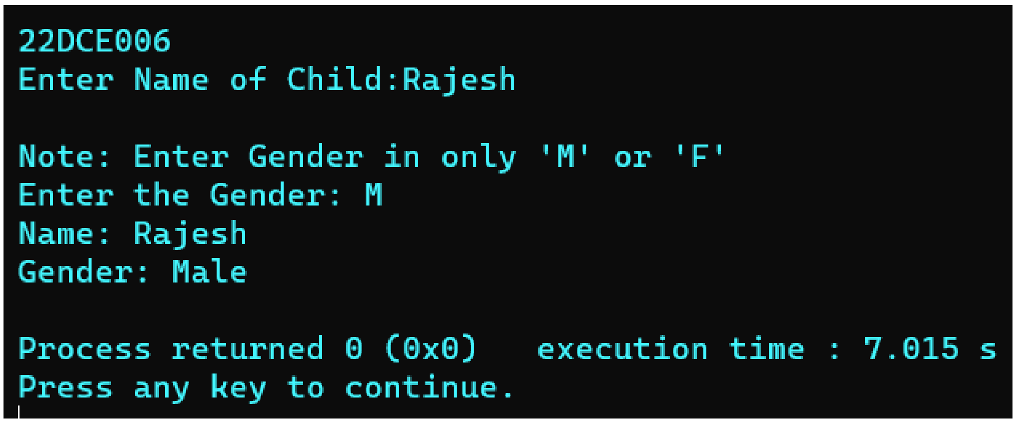
Code:

```
#include<iostream>

using namespace std;

class parent;  // forward declaration.

class child

{

   char name[20];

   char gender;

public :

   void getdata()

   {

      cout << "Enter Name of Child:";

      cin  >> name;

      cout<<"\n";

      cout << "Note: Enter Gender in only 'M' or 'F'" << endl;
```

```
    if(gender!='M' || gender!='m' || gender!='F' || gender!='f')
      {
              cout << "Enter the Gender: ";
              cin  >> gender;
      }
    }
    void print()
    {
       cout << "Name: " << name << endl;
       if(gender=='M' || gender=='m')
       {
          cout << "Gender: Male" << endl;
       }
       else
       {
          cout << "Gender: Female" << endl;
       }
    }
};
class parent
{
public :
  friend class child;
  void readchilddata(child x)
  {
    x.print();
  }
```

```
};
int main()
{
    cout<<"\n22DCE006\n";
    child a;
    parent m;
    a.getdata();
    m.readchilddata(a);
    return 0;
}
```

Output:



```
22DCE006
Enter Name of Child:Rajesh

Note: Enter Gender in only 'M' or 'F'
Enter the Gender: M
Name: Rajesh
Gender: Male

Process returned 0 (0x0)   execution time : 7.015 s
Press any key to continue.
```

CONCLUSION: We learnt to create one class, a friend of another class which can access all the data of the friend class.The entire class can also be taken as friend class for another class.

Practical-6.1

Write a C++ program having class time with data members: hr,min and sec. Define following member functions.

1) getdata() to enter hour, minute and second values

2) putdata() to print the time in the format 11:59:59

3) default constructor

4) parameterized constructor

Use 52 as default value for sec in parameterized constructor.

5) copy constructor

6) Destructor.

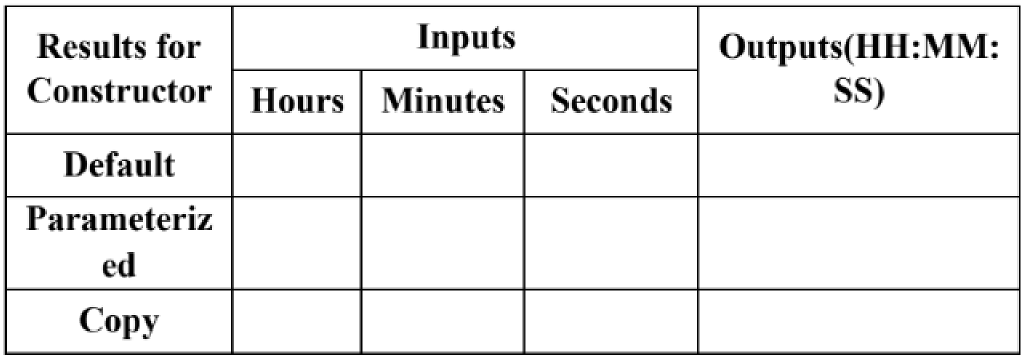
Use the concepts of default constructor, parameterized constructor,

Copy constructor, constructor with default arguments and

destructor.

Expected Output:

Fill the following table to showcase your outcome, also attach the screenshot of output.

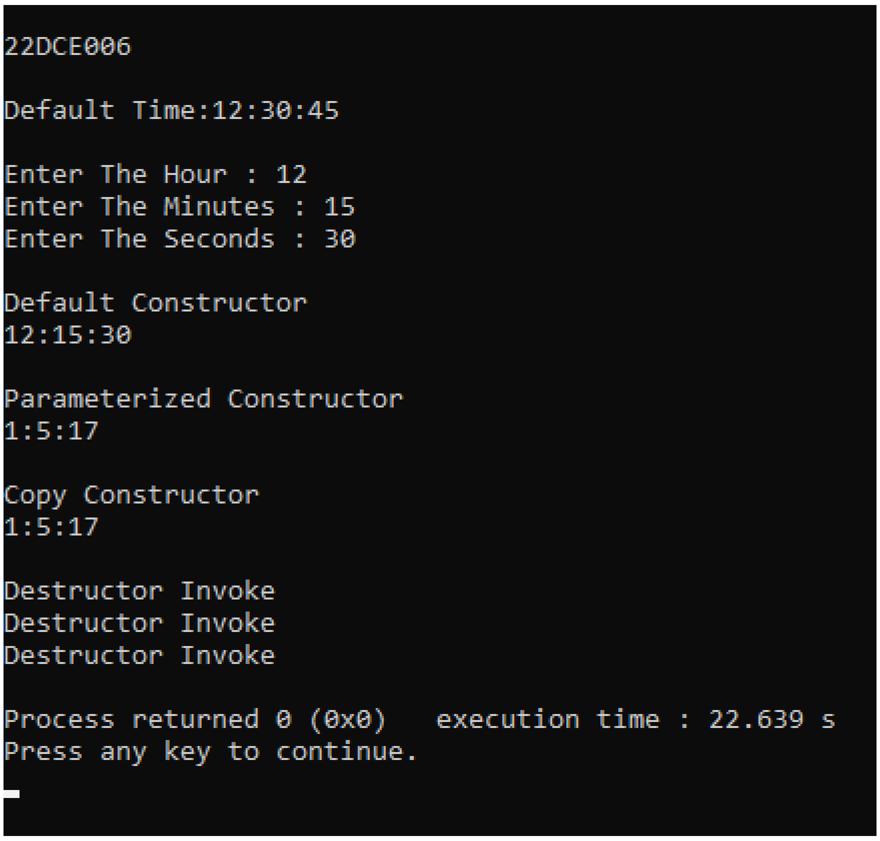


Results for Constructor	Inputs			Outputs(HH:MM: SS)
	Hours	Minutes	Seconds	
Default				
Parameterized				
Copy				

Code:

```
#include<iostream>
using namespace std;
class time
{
  int hr,Min,sec;
public:
 void getdata();
 void putdata();

ime()
 {
hr=10;
 Min=10;
 sec=10;
 }
 ~time()
{
cout<<"Destructor Invoke"<<endl;
 }
time(int a,intb,int c=52)
 {
hr=a;
 Min=b;
 sec=c;
 }
time(time& t)
 {
hr=t.hr;
 Min=t.Min;
 sec=t.sec;
 }
};
void time::getdata()
{
cout<<"Enter The Hour : ";
cin>>hr;
cout<<"Enter The Minutes : ";
cin>>Min;
cout<<"Enter The Seconds : ";
cin>>sec;
}
void time::putdata()
{
cout<<hr<<":"<<Min<<":"<<sec<<endl;
}
int main()
{
cout<<"Default Time:"<<endl;
 time t1;
 t1.putdata();
cout<<"\n";
 t1.getdata();
 t1.putdata();
 time t2(5,5);
 t2.putdata();
 time t3(t2);
 t3.putdata();
 return 0;
}
```

Output:



```
22DCE006

Default Time:12:30:45

Enter The Hour : 12
Enter The Minutes : 15
Enter The Seconds : 30

Default Constructor
12:15:30

Parameterized Constructor
1:5:17

Copy Constructor
1:5:17

Destructor Invoke
Destructor Invoke
Destructor Invoke

Process returned 0 (0x0)   execution time : 22.639 s
Press any key to continue.
```

Questions:

1. Differentiate Default, Parameterized and Copy constructor.

Ans) Default Constructors: Default constructor is the constructor which doesn’t take any argument. It has no parameters.

Parameterized Constructors: It is possible to pass arguments to constructors. Typically, these arguments help initialize an object when it is created. To create a parameterized constructor, simply add parameters to it the way you would to any other function. When you define the constructor’s body, use the parameters to initialize the object.

Copy Constructor: A copy constructor is a member function which initializes an object using another object of the same class. Detailed article on Copy constructor.

CONCLUSION: Default constructor: When we don’t pass any arguments to the constructor, it is called default constructor. Parameterized constructor: When we pass arguments to the constructor when calling, then it is called parameterized constructor. Copy constructor: When we pass arguments with the & operator, it is called copy constructor. Constructor with default arguments: When we don’t pass arguments but still want to get default values, then we use constructor with default arguments. Destructor: When we want to destroy the constructor, we use ~ (tilde)symbol to declare the destructor

Practical Set- 7

Practical-7.1

Create a class Number having int num as member. The class has input and output functions. Overload unary operator (++) such that it support N1=N2++ and N3=++N1 and Overload unary (-) such that it supports N3 = - N3. Also define default, parameterized and copy constructor for the class. Use the concept of Overloading Unary Operators.

Expected Output:

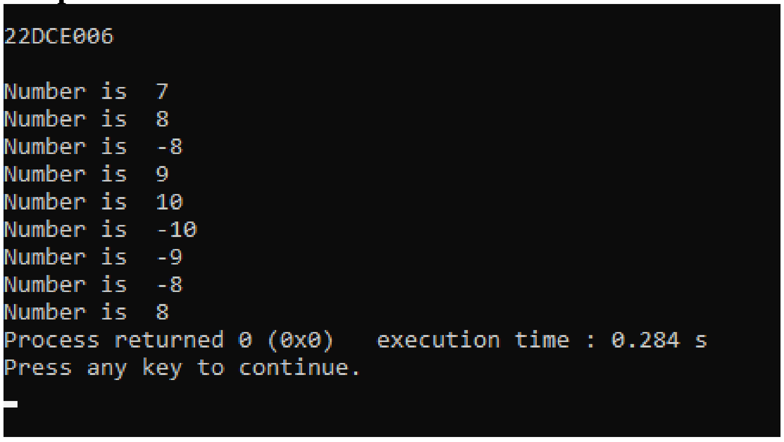
Fill up the below given table, according to the obtained output. Do it for two different inputs of your choice. Attach the screenshot of the output.

Code:

```
#include<iostream>
using namespace std;
class number
{
   public:
   int num;
   number():num(6){}
   void operator ++()
   {
      ++num;
   }
   void operator ++(int)
   {
      num++;
   }
   void operator -()
   {
      num=-num;
   }
   void display()
   {
      cout<<"\nNumber is  "<<num;
   }
   number(int x)
   {
      num=x;
   }
   number(const number&n2)
```

```
    {
        num=n2.num;
    }
};
int main()
{
  cout<<"\n22DCE006\n";
  number n1;
  ++n1;
  n1.display();
  n1++;
  n1.display();
  -n1;
  n1.display();

  number n2(8);
  ++n2;
  n2.display();
  n2++;
  n2.display();
  -n2;
  n2.display();

  number n3(n2);
  ++n3;
  n3.display();
  n3++;
  n3.display();
  -n3;
  n3.display();
}
```

Output:



```
22DCE006

Number is  7
Number is  8
Number is  -8
Number is  9
Number is  10
Number is  -10
Number is  -9
Number is  -8
Number is  8
Process returned 0 (0x0)   execution time : 0.284 s
Press any key to continue.
```

Question:

1. **Also explain use of nameless object in operator overloading.**

Ans) When we want to return an object from member function of class without creating an object, for this: we just call the constructor of class and return it to calling function and there is an object to hold the reference returned by constructor.This concept is known as nameless temporary objects, using this we can implement a C++ program for pre-increment operator overloading.

Conclusion:- When we overload unary operator with the help of member functions, we need not to pass any arguments at the time of calling. We can define the member function outside the class

Practical-7.2

Create a class complex having data members int real, img and member function to print data. Overload Unary operator (-) using Results for

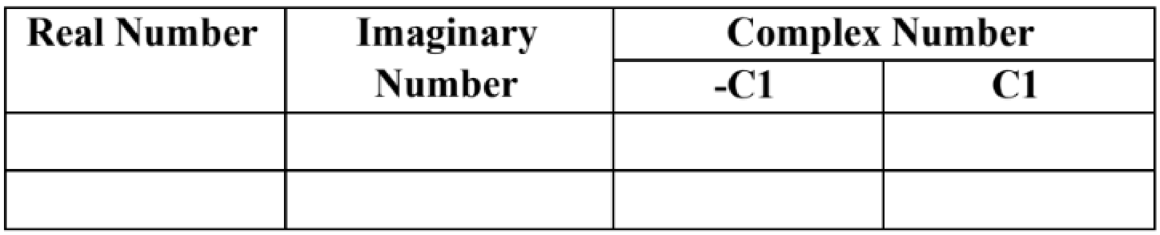
Constructor Inputs Outputs(HH:MM:Hours Minutes Seconds SS) Default Parameterized Copy Inputs Outputs Number Unary (++) N1=N2++ Unary

(++) N3=++N1 Unary (-) N3 = - N3 friend function such that it supports – C1 where C1 is the object of class complex. Also define default and parameterized constructor for the class. Use the concept of Overloading Unary Operators with friend function.

Expected Output:

Fill up the below given table, according to the obtained output. Do it for

two different inputs of your choice.



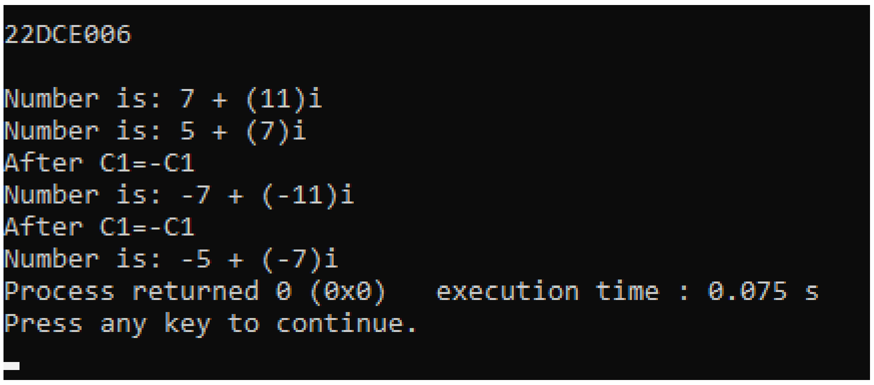
Real Number	Imaginary Number	Complex Number	
		-C1	C1

Code:

```
#include<iostream>
using namespace std;
class Complex
{
   int real,img;
public:
   Complex()
   {
      real=7;
      img=11;
```

```
    }
    Complex(int a, int b)
    {
        real=a;
        img=b;
    }
    void putdata()
    {
        cout<<"\nNumber is: "<<real<<" + ("<<img<<")i";
    }
    friend Complex operator-(Complex &ob1);
};

Complex operator-(Complex &ob1)
{
    Complex temp;
    temp.real=-ob1.real;
    temp.img=-ob1.img;
    return temp;
}

int main()
{
    cout<<"\n22DCE006\n";
    Complex C1;
    Complex C2(5,7);
    C1.putdata();
    C2.putdata();
    C1=-C1;
    cout<<"\nAfter C1=-C1";
    C1.putdata();
    C2=-C2;
    cout<<"\nAfter C1=-C1";
    C2.putdata();
}
```

Output:



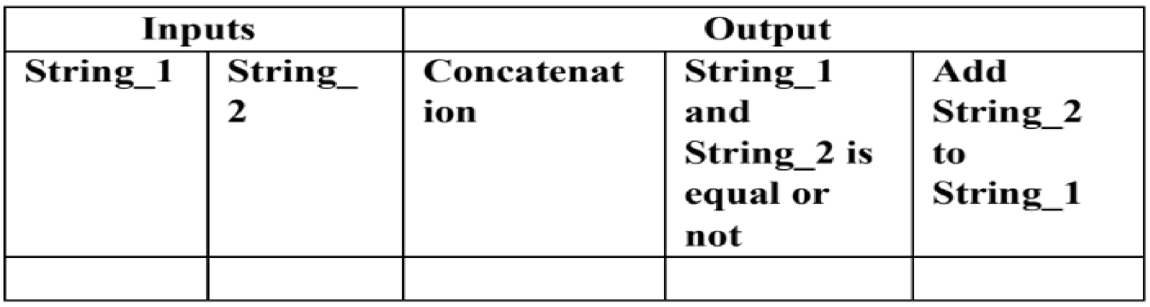
```
22DCE006

Number is: 7 + (11)i
Number is: 5 + (7)i
After C1=-C1
Number is: -7 + (-11)i
After C1=-C1
Number is: -5 + (-7)i
Process returned 0 (0x0)   execution time : 0.075 s
Press any key to continue.
```

Conclusion: When we overload unary operator with the help of friend functions, we need to pass only one argument at the time of calling. We can define the friend function outside the class.

Practical-7.3

Create a class String having character array. Class includes constructor and required member functions to get and display the object. Overload the operators +(s3=s1+s2), ==(s1<s2),+=(s1+=s2) for the class. Use the concept of Overloading Binary Operators.Expected Outputs:

Fill up the below given table, according to the obtained output. Attach the screenshot of the output.



Inputs		Output		
String_1	String_2	Concatenation	String_1 and String_2 is equal or not	Add String_2 to String_1

Code:

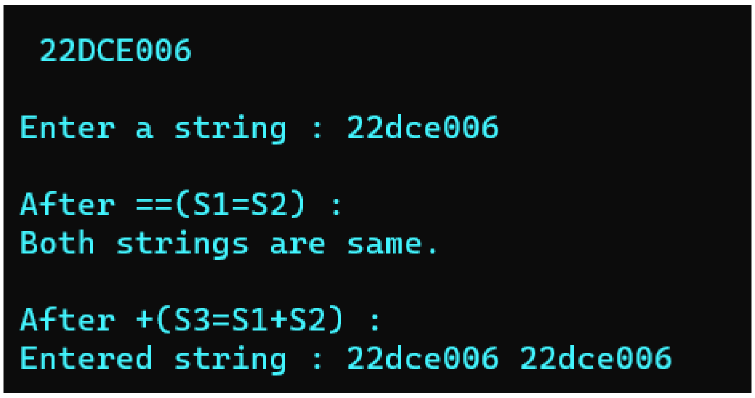
```
#include<iostream>

#include<string.h>

#include<cstring>

using namespace std;

class String

{

char str[20];

public:

String()

{ }

String(char str[])

{
```

```
strcpy(this->str,str);
}
void get()
{
cout << "\nEnter a string : ";
cin >> str;
}
void display()
{
cout << "Entered string : " << str << endl;
}
String operator +(String s2)
{
String s3;
strcat(str," ");
strcat(str,s2.str);
strcpy(s3.str,str);
return s3;
}
String operator +=(String s2)
{
strcat(str," ");
strcat(str,s2.str);
return str;
}
int operator==(String s2)
```

```
{
if(strcmp(str,"22dce006")==0)
return 1;
else
return 0;
}
};
int main()
{
cout <<"\n 22DCE006 \n";
String s1,s3;
s1.get();
String s2("22dce006");
cout << "\nAfter ==(S1=S2) : " << endl;
if(s1==s2)
cout << "Both strings are same." << endl;
else
cout << "Both strings are different." << endl;
s3=s1+s2;
cout << "\nAfter +(S3=S1+S2) : " << endl;
s3.display();
s1+=s2;
cout << "\nAfter +=(S1+=S2) : " << endl;
s1.display();

return 0;
```

}

Output:



```
 22DCE006

Enter a string : 22dce006

After ==(S1=S2) :
Both strings are same.

After +(S3=S1+S2) :
Entered string : 22dce006 22dce006
```

Practical-7.4

Create a class Celsius with float. Define appropriate member functions such that it support the statements: C1=30.5F; float temperature; temperature=C2; Use the concept of Type conversion from basic type to class type and class type to basic type. Expected Outcome:

Fill up the below given table, according to the obtained output. Attach the screenshot of the output.

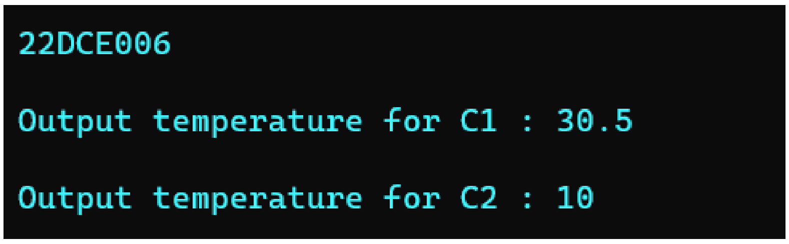
Value of C1	Value of Temperature

Code:

```
#include <iostream>
using namespace std;
class Celsius
{
float c;
public:
Celsius()
{
```

```
c=10;
}
Celsius(float k)
{
c=k;
}
operator float()
{
return(c);
}
void showdata()
{
cout << c << endl;
}
};
int main()
{
cout <<"\n 22DCE006\n";
Celsius c1,c2;
float temperature;
c1=30.5;
temperature=c2;
cout << endl << " Output temperature for C1 : ";
c1.showdata();
cout << endl << " Output temperature for C2 : ";
c2.showdata();
}
```

Output



```
22DCE006

Output temperature for C1 : 30.5

Output temperature for C2 : 10
```

Conclusion: When we the concept of type conversion from basic type to class type, we need to use constructor and in the conversion of class type to basic type we need to use operator keyword.

Practical-7.5

Create classes Celsius and Fahrenheit with float. Define appropriate member functions such that they support the statements in main( ): Celsius C1, C2=5.0; Fahrenheit F1, F2;F1=C2; C1=F2. Use the concepts of Type conversion from class type to class type. Write this Program in two ways. Define appropriate member function in class Celsius. Define appropriate member function in class Fahrenheit. Expected Output:

Fill up the below given table, according to the obtained output. Attach the screenshot of the output.

Input		Output	
Temperature in Celsius	Temperature in Fahrenheit	Celsius to Fahrenheit	Fahrenheit to Celsius

Code:

#include<iostream>

using namespace std;

class Fahrenheit;

class Celsius

{

public:

```
float ct;
Celsius()
{
ct=0;
}
Celsius(float temp)
{
ct = temp;
}
void getdata()
{
cout << "Enter temperature : ";
cin >> ct;
}
void putdata()
{
cout << "Temperature : " << ct << endl;
}
Celsius(const Fahrenheit &f);
};
class Fahrenheit
{
public:
float ft;
Fahrenheit()
{
```

```
ft = 0;
}
Fahrenheit(float temp)
{
ft = temp;
}
Fahrenheit(const Celsius &c)
{
ft=c.ct;
}
void putdata()
{
cout << "Temperature : " << ft << endl;
}
void getdata()
{
cout << "Enter temperature : ";
cin >> ft;
}
};
Celsius::Celsius(const Fahrenheit &f)
{
ct=f.ft;
}
int main()
{
```

cout <<"\n22DCE006\n";

Celsius c1, c2 = 5.0;

c2.putdata();

Fahrenheit f1, f2(13.2);

f1 = c2;

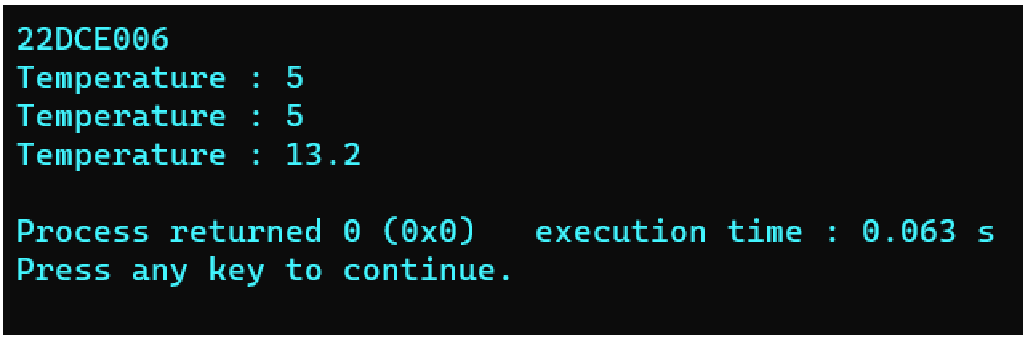
f1.putdata();

c1 = f2;

c1.putdata();

}

Output:



```
22DCE006
Temperature : 5
Temperature : 5
Temperature : 13.2

Process returned 0 (0x0)   execution time : 0.063 s
Press any key to continue.
```

Conclusion: We can do type conversion from one class type to another class type in the two different methods: 1) by using the concept of constructor and 2) with the help of operator keyword.

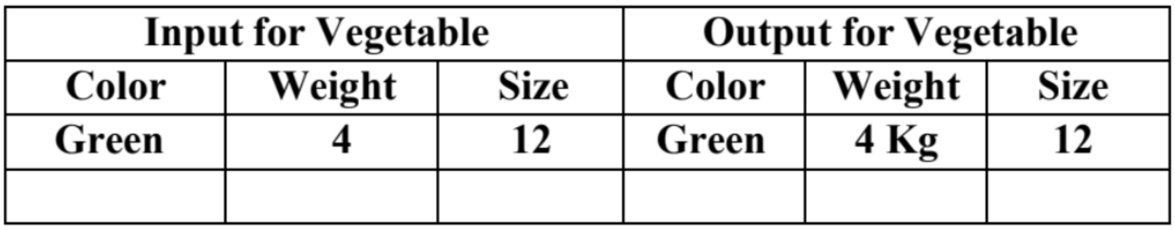
Practical Set- 8

Practical-8.1

Define a Base Class Vegetable having data member Color and member function getdata() which takes color as an input and putdata() which print the color as an output. Vegetable Class has one subclass named Tomato having data members weight and size and member function gtdata() which takes weight and size as an input and ptdata() which prints weight and size as output. Write a C++ Program which inherits the data of Vegetable class in Tomato class using Single Inheritance.

Expected Output:

Fill up the below given table, according to the obtained output. One sample is given, enter another one according to your choice. Attach the screenshot of the output.

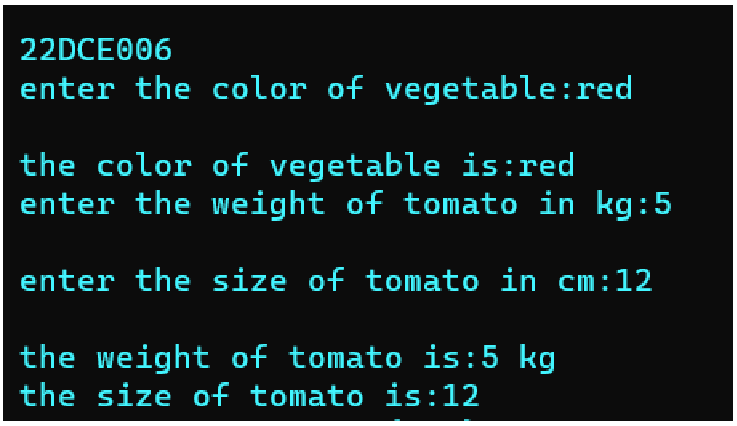


Input for Vegetable			Output for Vegetable		
Color	Weight	Size	Color	Weight	Size
Green	4	12	Green	4 Kg	12

Code:

```
#include<iostream>
#include<cstring>
using namespace std;
class vegetable
{
    char color[10];
    public:
        void get1data()
        {
            cout<<"enter the color of vegetable:";
            cin>>color;
        }
        void put1tdata()
        {
           cout<<"\nthe color of vegetable is:"<<color;
        }
};
class tomato:public vegetable
{
    int weight,size;
public:
    void getdata()
    {
        cout<<"\nenter the weight of tomato in kg:";
        cin>>weight;
        cout<<"\nenter the size of tomato in cm:";
        cin>>size;
    }
    void putdata()
    {
        cout<<"\nthe weight of tomato is:"<<weight<<" kg";
        cout<<"\nthe size of tomato is:"<<size;
           }
};
int main()
{
    cout<<"\n22DCE006\n";
    tomato  z1;
    z1.get1data();
    z1.put1data();
    z1.getdata();
    z1.putdata(); }
```

Output:



```
22DCE006
enter the color of vegetable:red

the color of vegetable is:red
enter the weight of tomato in kg:5

enter the size of tomato in cm:12

the weight of tomato is:5 kg
the size of tomato is:12
```

Conclusion: When we derive only one class from only one base class, then it is called single inheritance. In the single inheritance, derived class can access the properties and member functions of base class if it is declared publicly in the base class.

Practical-8.2

Write a program to create a class Medicine which stores type of medicine, name of company, date of manufacturing. Class Tablet isinherited from Medicine. Tablet class has name of tablet, quantityper pack, price of one tablet as members. Class Syrup is also inherited from Medicine and it has quantity per bottle, dosage unit as members. Both the classes contain necessary member functions for input and output data. Write a main ( ) that enter data for tablet and syrup, also display the data. Use the concepts of Hierarchical Inheritance. Expected Output:

Fill up the below given table, according to the obtained output. Attach

the screenshot of the output.

For Medicine type: Tablet

Company Name	Manufacturing date	Name of tablet	Quantity per pack	Price per tablet

For Medicine type: Syrup

Company Name	Manufacturing date	Quantity per Bottle	Dosage in ml

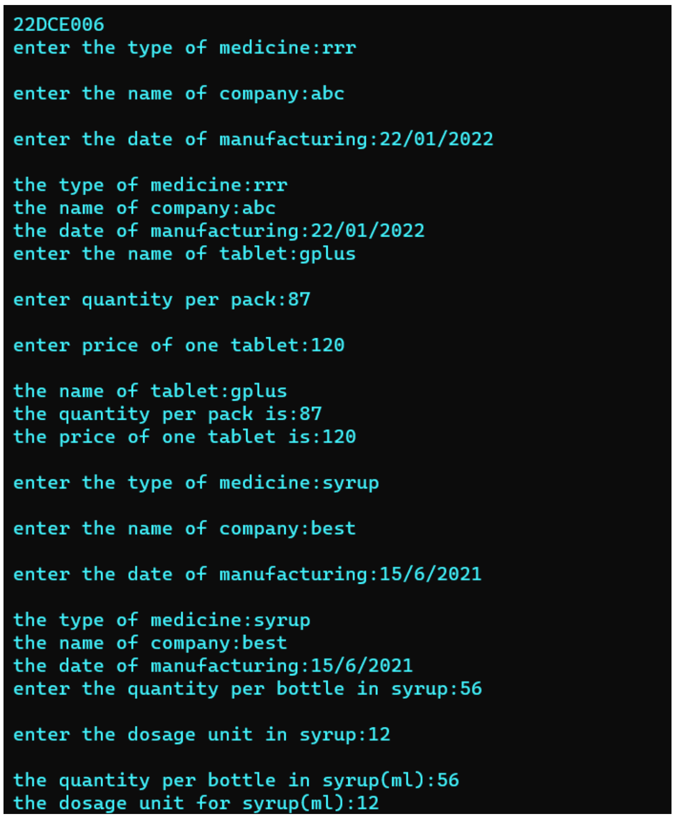
Code:

```
#include<iostream>
using namespace std;
class medicine
{
   string type_of_medicine,name_of_company,date_of_manufacturing;
   public:
   void getdata()
   {
      cout<<"\nenter the type of medicine:";
      cin>>type_of_medicine;
      cout<<"\nenter the name of company:";
      cin>>name_of_company;
      cout<<"\nenter the date of manufacturing:";
      cin>>date_of_manufacturing;
}
   void putdata()
   {
      cout<<"\nthe type of medicine:"<<type_of_medicine;
      cout<<"\nthe name of company:"<<name_of_company;
      cout<<"\nthe date of manufacturing:"<<date_of_manufacturing;
}
};
class tablet:public medicine
{
   string name_of_tablet,quantity_per_pack,price_of_one_tablet;
```

```
public:
   void gtdata()
   {
   cout<<"\nenter the name of tablet:";
   cin>>name_of_tablet;
   cout<<"\nenter quantity per pack:";
   cin>>quantity_per_pack;
   cout<<"\nenter price of one tablet:";
   cin>>price_of_one_tablet;
   }
   void ptdata()
   {
      cout<<"\nthe name of tablet:"<<name_of_tablet;
      cout<<"\nthe quantity per pack is:"<<quantity_per_pack;
      cout<<"\nthe price of one tablet is:"<<price_of_one_tablet;
   }
};
class syrup:public medicine
{
   string quantity_per_bottle,dosage_unit;
public:
   void gdata()
   {
      cout<<"\nenter the quantity per bottle in syrup:";
      cin>>quantity_per_bottle;
       cout<<"\nenter the dosage unit in syrup:";
```

```
        cin>>dosage_unit;
    }
    void pdata()
    {
        cout<<"\nthe quantity per bottle in syrup(ml):"<<quantity_per_bottle;
        cout<<"\nthe dosage unit for syrup(ml):"<<dosage_unit;
 }
};
int main()
{
    cout<<"\n22DCE006";
    tablet a1;
    a1.getdata();
    a1.putdata();
    a1.gtdata();
    a1.ptdata();
    cout<<endl;
    syrup b1;
    b1.getdata();
    b1.putdata();
    b1.gdata();
    b1.pdata();
}
```

Output:



```
22DCE006
enter the type of medicine:rrr

enter the name of company:abc

enter the date of manufacturing:22/01/2022

the type of medicine:rrr
the name of company:abc
the date of manufacturing:22/01/2022
enter the name of tablet:gplus

enter quantity per pack:87

enter price of one tablet:120

the name of tablet:gplus
the quantity per pack is:87
the price of one tablet is:120

enter the type of medicine:syrup

enter the name of company:best

enter the date of manufacturing:15/6/2021

the type of medicine:syrup
the name of company:best
the date of manufacturing:15/6/2021
enter the quantity per bottle in syrup:56

enter the dosage unit in syrup:12

the quantity per bottle in syrup(ml):56
the dosage unit for syrup(ml):12
```

Conclusion: When we derive more than one class from only one base class, then it is called hierarchical inheritance. In this inheritance, derived class can access the properties and member functions of the base class if it is declared publicly in the base class.

Practical-8.3

Create a Class alpha having data member: int x and one argument constructor which initializes the value of x. It also has member function which displays the value of x. Create another class beta which contains data member: float y and one argument constructor which initializes the value of y. It also has member function which displays the value of y. Create a Class Gamma which publicly inherits from class alpha and class beta and has two data members: int m, n and a constructor which passes argument to the base class constructor as well as initializes its own data members. Class Gamma also has member function to print the values of m and n. Write main function which creates object of class Gamma which passes values of base class constructor as well as derived class constructor.

Use the concept of Multiple Inheritance and Constructor in Derived Class.

Expected Output:

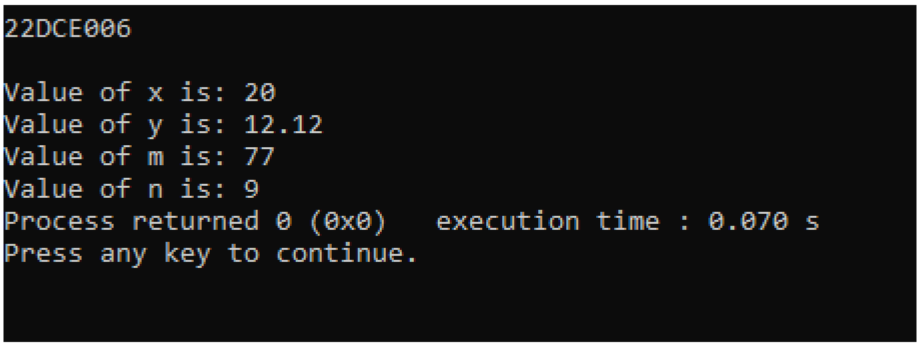
Fill up the below given table, according to the obtained output. Attach the screenshot of the output.

Value of x	Value of y	Value of m	Value of n

Code:

```
#include<iostream>
using namespace std;
class alpha
{
   protected:
   int x;
   public:
      alpha(int x1)
   {
      x=x1;
   }
   void show1()
   {
      cout<<"\nValue of x is: "<<x;
   }
};
class beta
{
   protected:
   float y;
   public:
      beta(float y1)
   {
      y=y1;
   }
    void show2()
   {
      cout<<"\nValue of y is: "<<y;
   }
};

class gamma:public alpha , public beta
{
int m,n;
      public:
   gamma(int m1,float n1,int m2,int n2):beta(n1),alpha(m1)
   {
      m=m2;
      n=n2;
   }
   void show3()
   {
      cout<<"\nValue of m is: "<<m;
      cout<<"\nValue of n is: "<<n;
   }
};
int main()
{
   cout<<"\n22DCE006\n";
   gamma g(20,12.12,77,9);
   g.show1();
   g.show2();
   g.show3();
}
```

Output:



```
22DCE006

Value of x is: 20
Value of y is: 12.12
Value of m is: 77
Value of n is: 9
Process returned 0 (0x0)   execution time : 0.070 s
Press any key to continue.
```

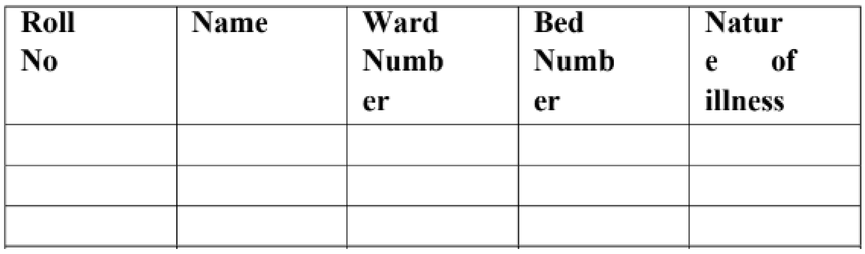
Conclusion: When we derive only one class from more than one base class, then it is called multiple inheritance. In this inheritance, derived class can access the properties and member functions of both of the base classes when it is declared publicly in the base classes.

Practical-8.4

Define a class Hospital having rollno and name as data members and member function to get and print data. Derive a class Ward from class Hospital having data members: ward number and member function to get and print data. Derive another class Room from Hospital having data member bed number and nature of illness and member function to get and print data. Derive class Patient from Class Ward and Class Room. In main () declare 5 object of Class Patient and get and display all the information. Use the concept of Virtual Base Class and Hybrid Inheritance Expected Output:

Fill up the below given table, according to the inputted data for 5

patients. Attach the screenshot of the output.



Roll No	Name	Ward Number	Bed Number	Nature of illness

Code:

```
#include <iostream>

using namespace std;

class Hospital

{

int roll_no;

char name[50];

public :
```

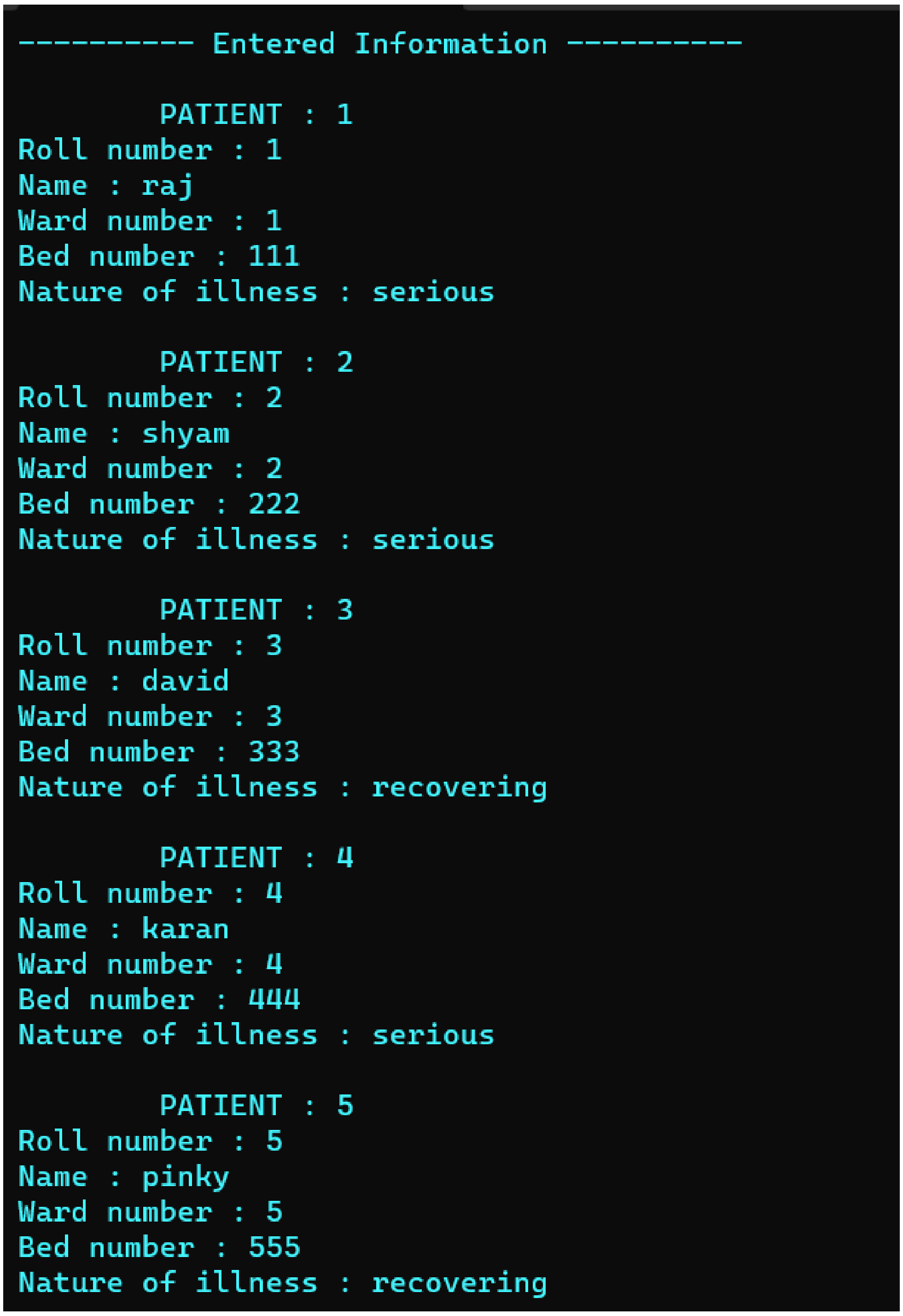
```
void geth()
{
cout << "Roll number : ";
cin >> roll_no;
cout << "Name : ";
cin >> name;
}
void puth()
{
cout << "Roll number : " << roll_no << endl;
cout << "Name : " << name << endl;
}
};
class Ward:public virtual Hospital
{
int ward_no;
public :
void getw()
{
cout << "Ward number : ";
cin >> ward_no;
}
void putw()
{
cout << "Ward number : " << ward_no << endl;
}
```

```
};
class Room:public virtual Hospital
{
int bed_no;
char nature[50];
public :
void getr()
{
cout << "Bed number : ";
cin >> bed_no;
cout << "Nature of illness : ";
cin >> nature;
}
void putr()
{
cout << "Bed number : " << bed_no << endl;
cout << "Nature of illness : " << nature << endl;
}
};
class Patient:public Ward, public Room
{
public :
void getdata()
{
geth();
getw();
```

```
getr();
}
void display()
{
puth();
putw();
putr();
}
};
int main()
{
cout <<"\n22DCE006 \n";
Patient p[5];
int x=1;
for(int i=0;i<=4;i++)
{
cout << endl << "\tPATIENT : " << x << endl;
p[i].getdata();
x++;
}
cout << endl << endl << "---------- Entered Information ----------" << endl;
int y=1;
for(int i=0;i<=4;i++)
{
cout << endl << "\tPATIENT : " << y << endl;
p[i].display();
```

```
y++;
}
}
```

Output:



```
---------- Entered Information ----------

        PATIENT : 1
Roll number : 1
Name : raj
Ward number : 1
Bed number : 111
Nature of illness : serious

        PATIENT : 2
Roll number : 2
Name : shyam
Ward number : 2
Bed number : 222
Nature of illness : serious

        PATIENT : 3
Roll number : 3
Name : david
Ward number : 3
Bed number : 333
Nature of illness : recovering

        PATIENT : 4
Roll number : 4
Name : karan
Ward number : 4
Bed number : 444
Nature of illness : serious

        PATIENT : 5
Roll number : 5
Name : pinky
Ward number : 5
Bed number : 555
Nature of illness : recovering
```

CONCLUSION:

The process of applying two or more types of inheritance to design a program is called hybrid inheritance. When all three types of inheritance, namely multilevel, multiple and hierarchical inheritance are involved, then we need to use virtual keyword to make classes virtual to avoid ambiguity in the program.

Practical-8.5

Create a class shape having data member shape_name and member function to get and print shape_name. Derive a Class Circle which is inherited publicly from class shape and having data members radius of a circle and member function to get and print radius of a circle. Derive a Class Area which is inherited publicly from Class Circle and having data members area_of_circle and member function display () which displays area of a circle. Use object of class Area in main () function and get and display all the information. Use the concepts of Multilevel Inheritance.

Expected output:

Fill up the below given table, according to the given inputs and obtained outputs. Attach the screenshot of the output.

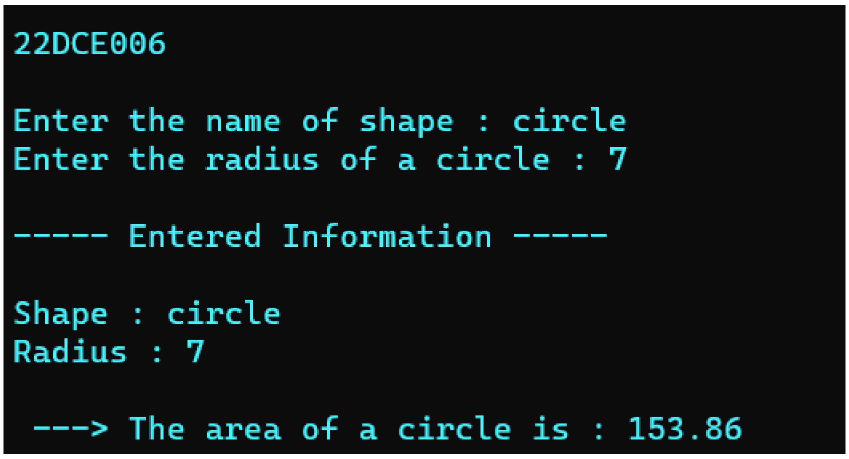
Inputs		Output
Name of Shape	**Radius of Circle**	**Area of Circle**

Code:

```
#include <iostream>
using namespace std;
class Shape
{
string shape_name;
public:
void str_get()
{
cout << endl << "Enter the name of shape : ";
cin >> shape_name;
}
void str_put()
{
cout << endl << "Shape : " << shape_name << endl;
}
};
class Circle:public Shape
{
protected:
float radius_circle;
public:
void c_get()
{
str_get();
cout << "Enter the radius of a circle : ";
cin >> radius_circle;
}
void c_put()
{
str_put();
cout << "Radius : " << radius_circle << endl;
}
};
class Area:public Circle
{
float area_of_circle;
public :
void display()
```

```
{
area_of_circle = (3.14*(radius_circle*radius_circle));
cout << endl << " ---> The area of a circle is : " << area_of_circle << endl;
}
};
int main()
{
cout << "\n22DCE006\n";
Area a1;
a1.c_get();
cout << endl << "----- Entered Information -----" << endl;
a1.c_put();
a1.display();
}
```

Output:



```
22DCE006

Enter the name of shape : circle
Enter the radius of a circle : 7

----- Entered Information -----

Shape : circle
Radius : 7

 ---> The area of a circle is : 153.86
```

Conclusion:

The process of deriving a class from another derived class is called multilevel inheritance. When we want to use the already defined functions in the base classes we use the concept of multilevel inheritance.

Practical Set- 9

Practical-9.1

What is the output of the following code: a) Pointer to objects

```
#include<iostream>
using namespace std;
class product
{ int code;
float price;
public:
void getdata(int a, float b)
{
code=a;
price=b;
}
void show()
{
cout<<"Code: "<<code<<endl;
cout<<"Price: "<<price<<endl;
}
};
int main(){
product * p = new product;
product *d = p;
int x,i;
float y;
cout<<"Input code and price for product: ";
```

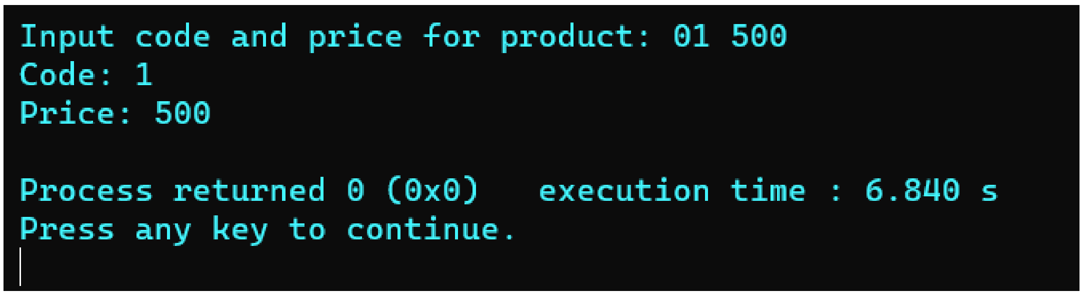
cin>>x>>y;

p->getdata(x,y);

d->show();

}

Attach the screenshot of the received output and write your explanation about it in few words.



```
Input code and price for product: 01 500
Code: 1
Price: 500

Process returned 0 (0x0)   execution time : 6.840 s
Press any key to continue.
```

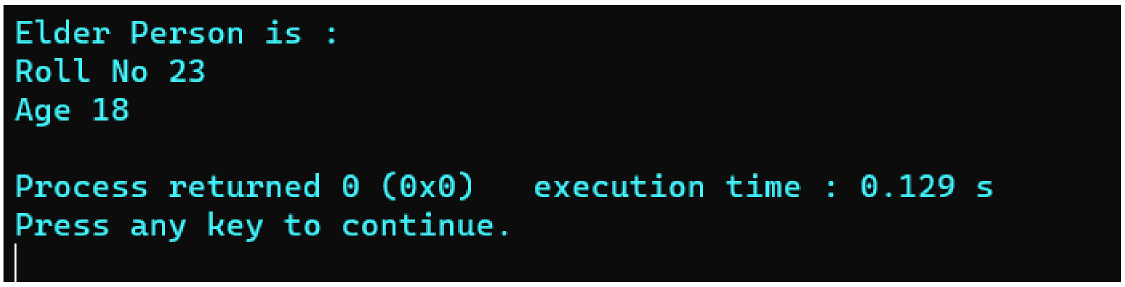
Conclusion: When we use the concept of pointer to object, we can call the member functions by the -> operator. It can call the member functions inside the class directly.

Practical-9.2

What is the output of the following code: b) this Pointer

```
#include<iostream>
using namespace std;
class student
{
int roll_no;
float age;
public:
student(int r, float a)
{
roll_no = r;
age = a;
}
student & greater (student & x)
{
if(x.age>=age)
return x;
else
{
return *this;
}
void display()
{
cout<<"Roll No "<<roll_no<<endl;
cout<<"Age "<<age<<endl;
}
};
int main()
{
student s1 (23,18),s2 (30,20),s3 (45,16);
student s = s1.greater(s3);
cout<<"Elder Person is :"<<endl;
s.display();
}
```

Attach the screenshot of the received output and write your explanation about it in few words.



```
Elder Person is :
Roll No 23
Age 18

Process returned 0 (0x0)   execution time : 0.129 s
Press any key to continue.
```

Conclusion:

Here this pointer is used to invoke member function with the help of current object. Using this pointer we can call the functions inside class with the use of current object.

Practical-9.3

c) Pointers to Derived Objects

```
#include<iostream>

using namespace std;

class BC

{

public:

int b;

void show()
```

```
{
cout<<"b = "<<b<<endl;
}
};
class DC : public BC
{
public:
int d;
void show()
{
cout<<"b = "<<b<<endl;
cout<<"d = "<<d<<endl;
}
};
int main()
{
BC *bptr;
BC base;
bptr = &base;
bptr->b = 100;
cout<<"bptr poins to base objects"<<endl;
bptr->show();
DC derived;
bptr = &derived;
bptr->b = 200;
/*bptr->b = 300;*/ // wont work
```

cout<<"bptr now points to derived object"<<endl;

bptr->show();

DC *dptr;

dptr=&derived;

dptr->d=300;

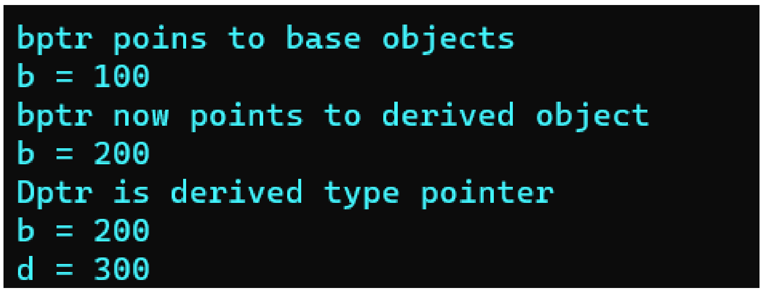
cout<<"Dptr is derived type pointer"<<endl;

dptr->show();

return 0;

}

Attach the screenshot of the received output and write your explanation about it in few words.



```
bptr poins to base objects
b = 100
bptr now points to derived object
b = 200
Dptr is derived type pointer
b = 200
d = 300
```

Conclusion: In the concept of pointer to derived objects, when we call the member function with help of pointer, it will give first preference to the class of which it is made a pointer.

Practical-9.4

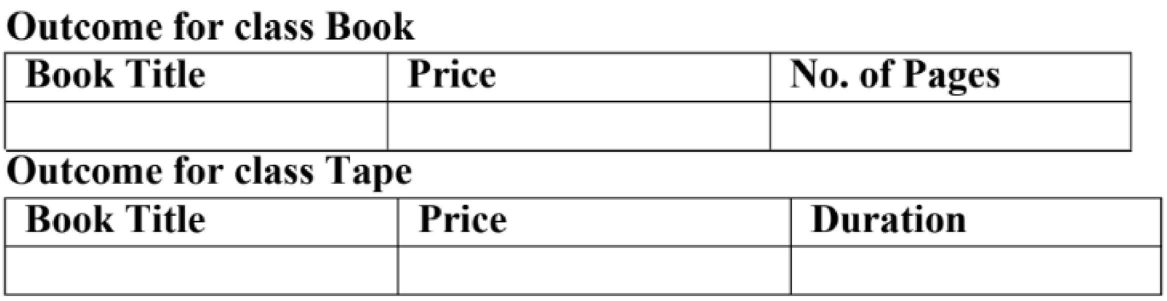
Create a class Media that stores the title (a string) and price (float). Class Media has two argument constructor which initializes data members of class Media. Also declare a virtual function display () in Class Media. From the class Media derive two classes: Class book, which contains data member page count (int): and Class tape, which contains data member playing time in minutes (float). Both Class book and Class tape should have a constructor which initializes base class constructor as well as its own data members and display ( ) function which displays book details and tape

details respectively. Write a main ( ) to test book and tape classes by creating instances of them, asking the user to fill data and displaying them.

Use the concept of Virtual function and Constructor in Derived Class.

Expected Output:

Fill up the below given table, according to the obtained output. Attach the screenshot of the output.



Outcome for class Book

Book Title	Price	No. of Pages

Outcome for class Tape

Book Title	Price	Duration

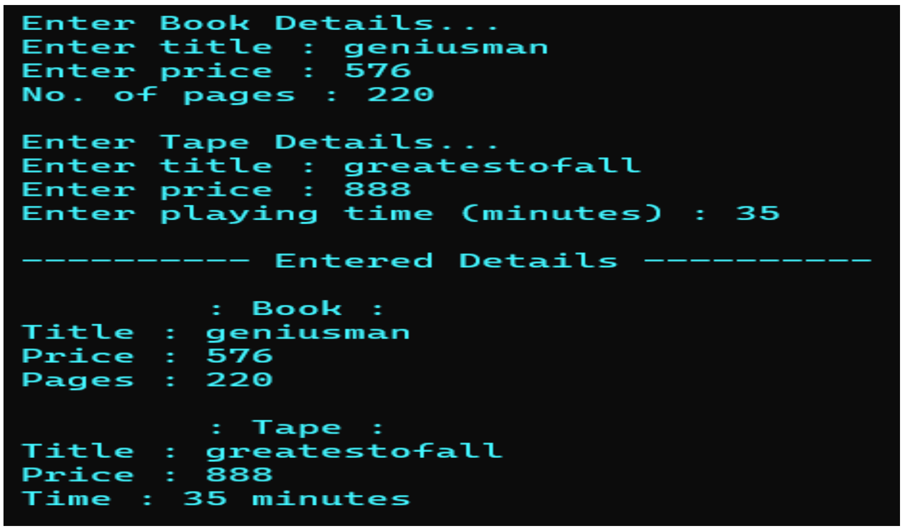
Code:

```
#include <iostream>

#include <cstring>

using namespace std;

class Media

{

protected:

string title;

float price;

public:

Media() // default constructor

{}

Media(string s, float p) // parameterized constructor

{

title=s;

price=p;
}
virtual void display() // virtual function
{
cout << "Title : " << title << endl;
cout << "Price : " << price << endl;
}
};
class Book:public Media
{
int page;
public:
Book(string s, float p, int a):Media(s,p)
{
page=a;
}
void display()
{
cout << "Title : " << title << endl;
cout << "Price : " << price << endl;
cout << "Pages : " << page << endl;
}
};
class Tape:public Media
{
float time;
```

```
public:
Tape(string s, float p, int b):Media(s,p)
{
time=b;
}
void display()
{
cout << "Title : " << title << endl;
cout << "Price : " << price << endl;
cout << "Time : " << time << " minutes" << endl;
}
};
int main()
{
cout<<"\n22DCE006\n";
string book_title,tape_title;
float book_price, tape_price, time;
int page;
cout << endl << "Enter Book Details..." << endl;
cout << "Enter title : ";
cin >> book_title;
cout << "Enter price : ";
cin >> book_price;
cout << "No. of pages : ";
cin >> page;
```

```
cout << endl << "Enter Tape Details..." << endl;
cout << "Enter title : ";
cin >> tape_title;
cout << "Enter price : ";
cin >> tape_price;
cout << "Enter playing time (minutes) : ";
cin >> time;
Book book1(book_title, book_price, page);
Tape tape1(tape_title, tape_price, time);
Media *m1, *m2;
m1=&book1;
m2=&tape1;
cout << endl << "---------- Entered Details ----------" << endl;
cout << endl << "\t: Book : " << endl;
m1->display();
cout << endl << "\t: Tape : " << endl;
m2->display();
}
```

Output:



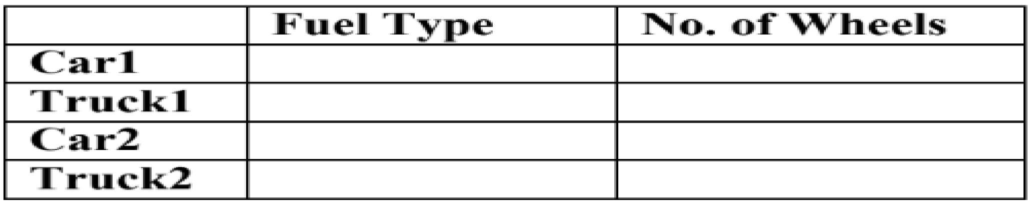
```
Enter Book Details...
Enter title : geniusman
Enter price : 576
No. of pages : 220

Enter Tape Details...
Enter title : greatestofall
Enter price : 888
Enter playing time (minutes) : 35

---------- Entered Details ----------

        : Book :
Title : geniusman
Price : 576
Pages : 220

        : Tape :
Title : greatestofall
Price : 888
Time : 35 minutes
```

Conclusion: When we call the member function with help of pointer of a particular class, it will give first preference to the class of which it is made a pointer. If that class has normal member function then it will invoke that else if it has virtual function then it will invoke the derived class function.

Practical-9.5

Create an Abstract class vehicle having average as data and pure virtual function getdata() and putdata(). Derive class car and truck from class vehicle having data members: fuel type (petrol, diesel, CNG) and no of wheels respectively. Write a main ( ) that enters the data of two cars and a truck and display the details of them. Use the concept of Abstract Base class and Pure Virtual functions. Expected Output:

Fill up the below given table, according to the obtained output. Attach the screenshot of the output.



	Fuel Type	No. of Wheels
Car1		
Truck1		
Car2		
Truck2		

Code:

```
#include <iostream>
using namespace std;
class Vehicle
{
protected:
float average;
public :
virtual void getdata(){}
virtual void putdata(){}
};
```

```
class Car:public Vehicle
{
string car_fuel;
int car_wheels;
public :
void getdata()
{
cout << "Enter The Information for Car..." << endl;

cout << endl << "Fuel type : ";
cin >> car_fuel;
cout << "No. of wheels : ";
cin >> car_wheels;
}
void putdata()
{
cout << "Fuel type : " << car_fuel << endl;
cout << "No. of wheels : " << car_wheels << endl;
}
};
class Truck:public Vehicle
{
string truck_fuel;
int truck_wheels;
public:
void getdata()
```

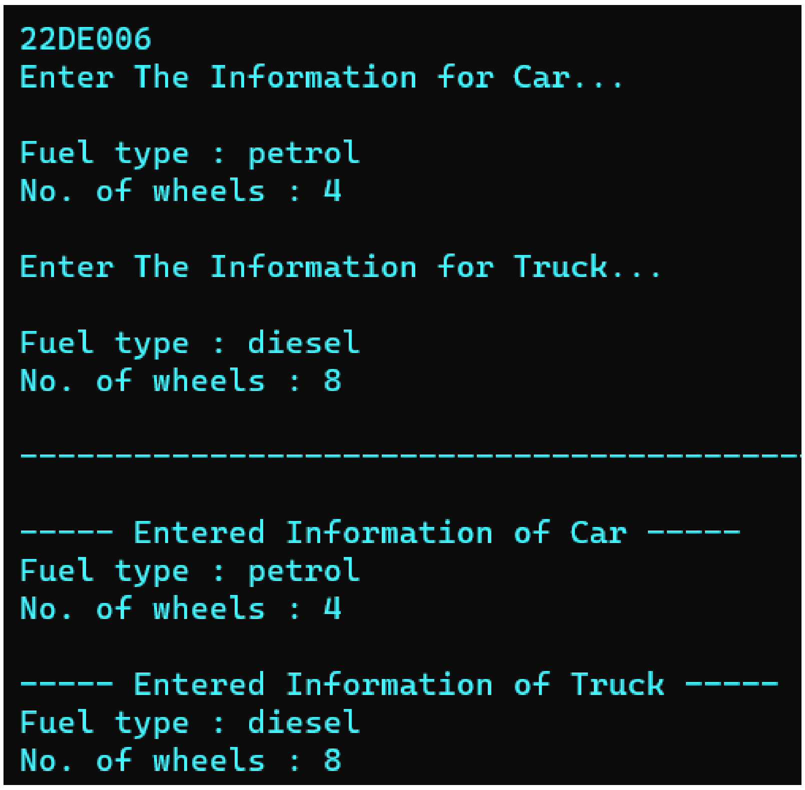
```
{
cout << endl << "Enter The Information for Truck..." << endl;
cout << endl << "Fuel type : ";
cin >> truck_fuel;
cout << "No. of wheels : ";
cin >> truck_wheels;
}
void putdata()
{
cout << "Fuel type : " << truck_fuel << endl;
cout << "No. of wheels : " << truck_wheels << endl;
}
};
int main()
{
cout<<"\n22DE006\n";
Vehicle *v1,*v2;
Car c1;
Truck t1;
v1=&c1;
v2=&t1;
v1->getdata();
v2->getdata();
cout << endl << "-----------------------------------------------" << endl;
cout << "\n----- Entered Information of Car -----" << endl;
v1->putdata();
```

cout << "\n----- Entered Information of Truck -----" << endl;

v2->putdata();

}

Output:



```
22DE006
Enter The Information for Car...

Fuel type : petrol
No. of wheels : 4

Enter The Information for Truck...

Fuel type : diesel
No. of wheels : 8

------------------------------------------


----- Entered Information of Car -----
Fuel type : petrol
No. of wheels : 4

----- Entered Information of Truck -----
Fuel type : diesel
No. of wheels : 8
```

Conclusion: If a class contains at least one pure virtual function, then it is called an abstract class. We can define pure virtual function as follows: virtual void func()=0;

Practical Set- 10

Practical-10.1

What is the output of the following code related to ios format functions, get() and put() functions and getline() and write() functions?

(a)

```
#include<iostream>
using namespace std;
int main()
{
char s[12]="ABC_DEF_GHI";
cout.write(s,9);
int x=12345;
cout.fill('*');
cout.width(10);
cout<<endl<<x;
return 0;
}
```

(b)

```
int main()
{
int a,b;
a = (b = 50) +10;
cout<<"a = "<<a<<endl; cout<<"b = "<<b<<endl;
float x=23.4;
cout.fill('*');
cout.width(10);
cout<<x<<endl;
float y=54.4;
cout.setf(ios::showpos);
cout<<y<<endl;
return 0;
}
```

(c)

```
#include<iostream>
using namespace std;
int main()
{
int count =0;
char c;
cout<<"INPUT TEXT\n";
cin.get(c);
while(c!='\n')
{
cout.put(c);
count++;
cin.get(c);
}
cout<<"\n Number of charaters = "<<count<<"\n";
return 0;
}
```

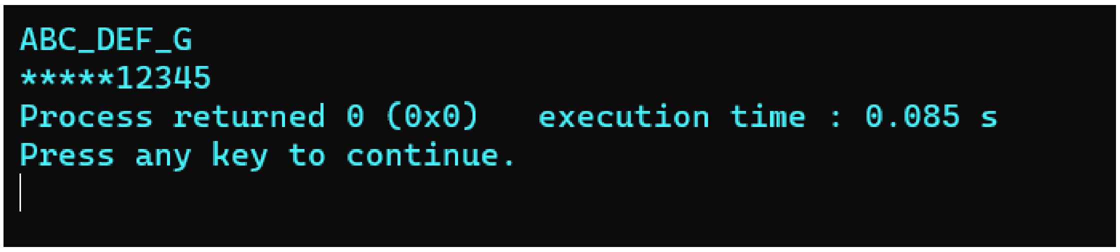
(d)

```
#include<iostream>
using namespace std;
int main()
{
char name[20];
cout<<"Enter first name then white space and then last name of a
person: ";
cin>>name;
cout<<"Person Name : "<<name<<endl;
cout<<"Enter first name then white space and then last name of a
person: ";
cin.getline(name,10);
cout.write(name,7);
cout<<"Again Enter first name then white space and then last
name of a person: ";
cin.getline(name,13);
cout.write(name,11);
return 0;
}
```

Attach the screenshot of the output and explain them.

Outputs:

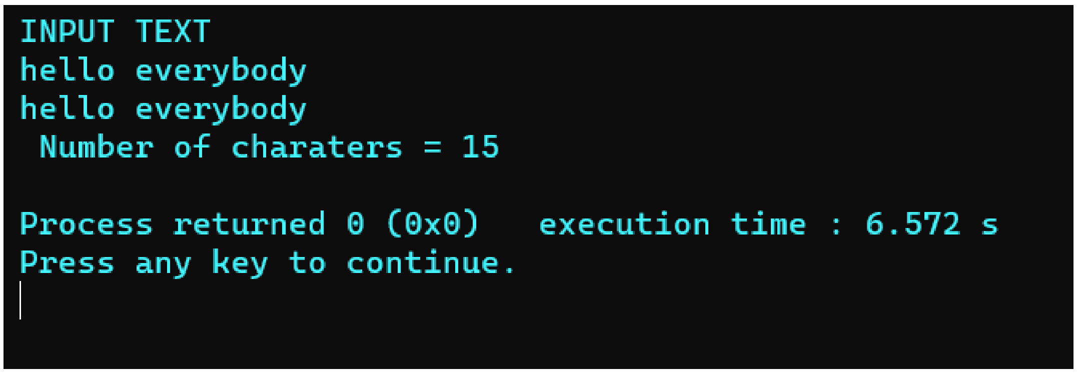
a)



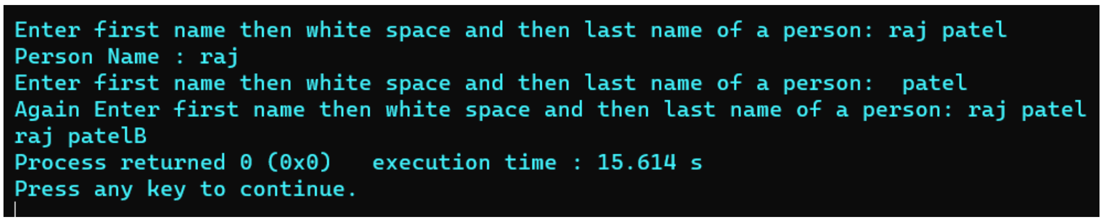
```
ABC_DEF_G
*****12345
Process returned 0 (0x0)   execution time : 0.085 s
Press any key to continue.
```

b) No output as header files are not included.

c)


```
INPUT TEXT
hello everybody
hello everybody
 Number of charaters = 15

Process returned 0 (0x0)   execution time : 6.572 s
Press any key to continue.
```

d)


```
Enter first name then white space and then last name of a person: raj patel
Person Name : raj
Enter first name then white space and then last name of a person:  patel
Again Enter first name then white space and then last name of a person: raj patel
raj patelB
Process returned 0 (0x0)   execution time : 15.614 s
Press any key to continue.
```

Practical-10.2

Write a program which demonstrates how to create user-defined Manipulators. Attach the screenshot of the output.

Code:

```
#include<iostream>

#include<iomanip>

using namespace std;

ostream&curr(ostream&ostrobj)

{

cout<< fixed <<setprecision(2);

cout<< "Rs. ";

return ostrobj;

}

int main()
```

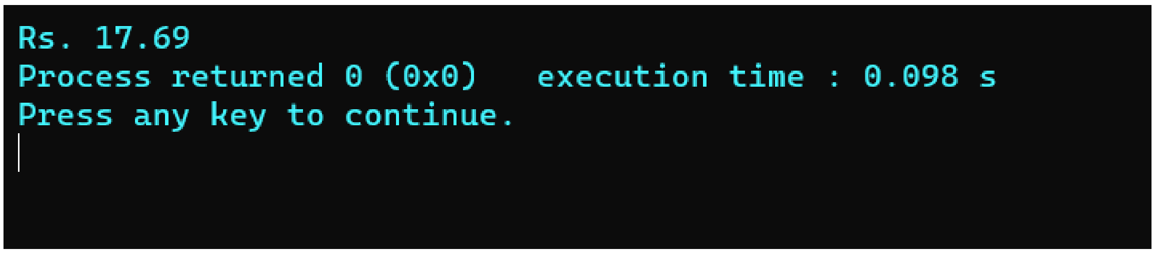
{

float amt = 17.6877;

cout<<curr<< amt;

}

Output:



```
Rs. 17.69
Process returned 0 (0x0)   execution time : 0.098 s
Press any key to continue.
```

Practical Set- 11

Practical-11.1

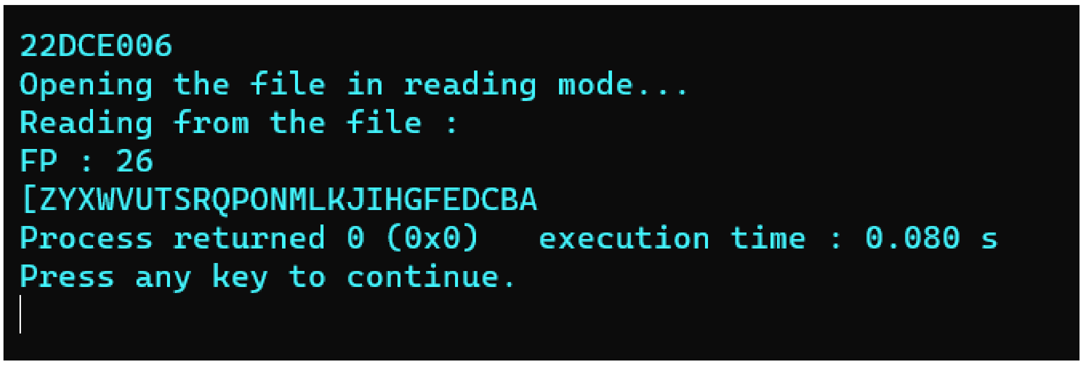
Write a program that creates a text file that contains ABC...Z. A program should print the file in reverse order on the screen. i.e. ZYX...BA. Use concept of Opening the file using constructor and open() function. Use all error handling functions like eof() , fail() , bad() , good() and functions for manipulation of file pointer like seekg() and tellg().

Attach the screenshot of the output.

Code:

```
#include<iostream>
#include<iomanip>
#include<fstream>
using namespace std;
int main()
{
cout<<"\n22DCE006\n";
ofstream fp("2dce1.txt");
if(fp.good())
{
for(char i=65;i<=91;i++)
{
fp.put(i);
}
}
fp.close();
ifstream ip("2dce1.txt");
cout<<"Opening the file in reading mode...";
if(!ip.bad() && !ip.fail() && !ip.eof())
{
cout << "\nReading from the file : ";
ip.seekg(-1,ios::end);
cout << "\nFP : " << ip.tellg() << endl;
char ch;
while(ip.tellg()>=0)
{
if(!ip.eof())
{
ip.get(ch);
cout << ch;
ip.seekg(-2,ios::cur);
}
}
}
ip.close();
}
```

Output:



```
22DCE006
Opening the file in reading mode...
Reading from the file :
FP : 26
[ZYXWVUTSRQPONMLKJIHGFEDCBA
Process returned 0 (0x0)   execution time : 0.080 s
Press any key to continue.
```

Conclusion: From this program, I learnt about handling the files and the proper uses of the functions manipulators for pointers like seekg(), tellg().Also, I learnt the file error handling functions such as eof(), fail(), bad(), good()

Practical-11.2

Write a program that creates a binary file and input height in float for the five students. Display the content of the file with two precision. Use the concept of Write() and read() functions for handling data in binary form. Attach the screenshot of the output.

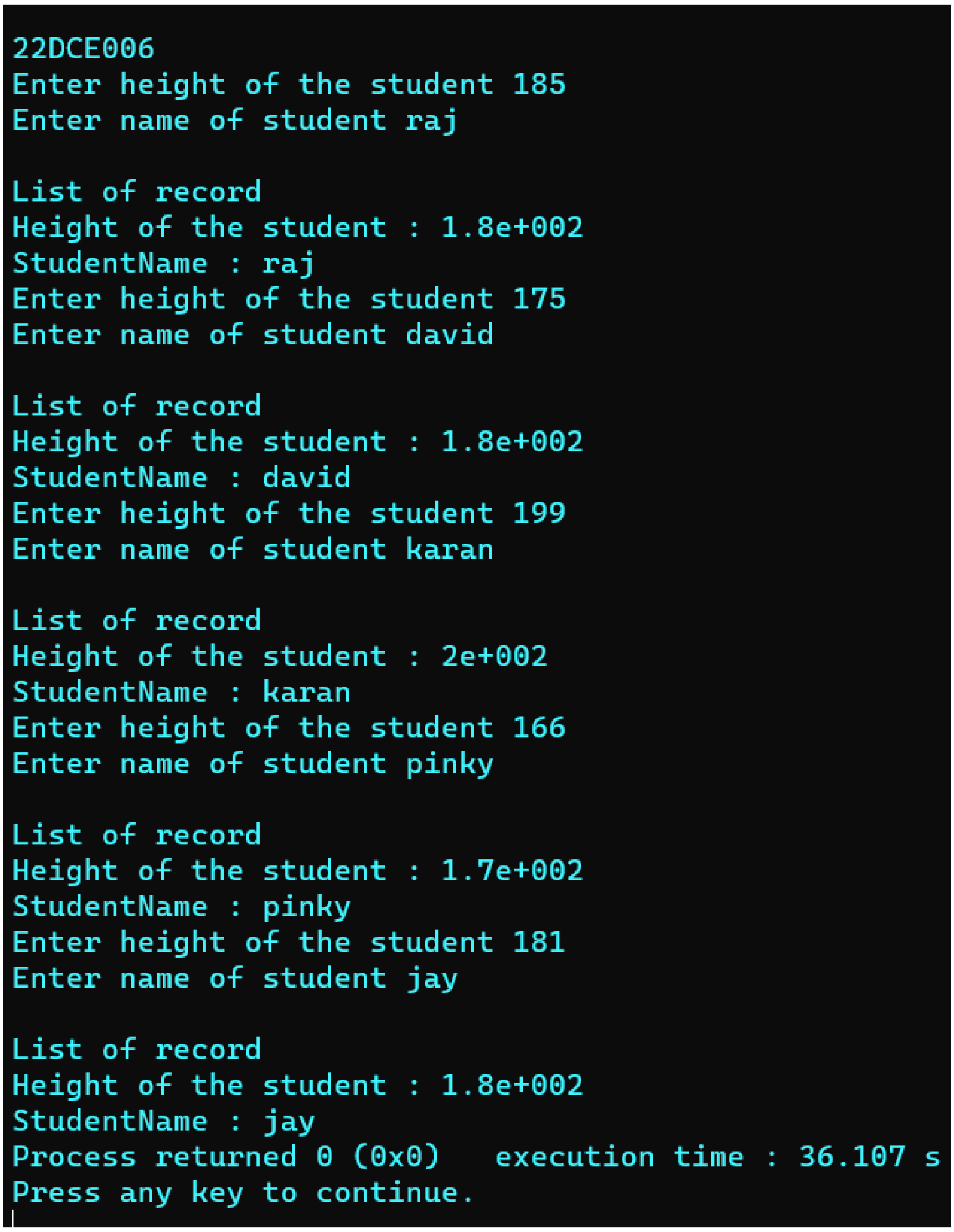
Code:

```
#include<iostream>

#include<fstream>

#include<cstdio>

#include<iomanip>

using namespace std;

class Student

{

   float height;

   char name[50];

   public:

   void setData()

   {

      cout<< "\nEnter height of the student ";

      cin>> height;

      cout<< "Enter name of student ";

      cin>> name;

   }

   void showData()
```

```
    {
        cout<< "\nHeight of the student : " <<setprecision(2)<< height;
        cout<< "\nStudentName : " << name;
    }
};
void write_record()
{
    ofstream outFile;
    outFile.open("student.dat", ios::binary);
    Student obj;
    obj.setData();
    outFile.write((char*)&obj, sizeof(obj));
    outFile.close();
}
void display()
{
    ifstream inFile;
    inFile.open("student.dat", ios::binary );
    Student obj;
    while(inFile.read((char*)&obj, sizeof(obj)))
    {
        obj.showData();
    }
    inFile.close();
}
int main()
```

```
{
   cout<<"\n22DCE006";
   for(int i = 1; i<= 5; i++)
   {
      write_record();
      cout<< "\nList of record";
      display();
  }
}
```

Output:



```
22DCE006
Enter height of the student 185
Enter name of student raj

List of record
Height of the student : 1.8e+002
StudentName : raj
Enter height of the student 175
Enter name of student david

List of record
Height of the student : 1.8e+002
StudentName : david
Enter height of the student 199
Enter name of student karan

List of record
Height of the student : 2e+002
StudentName : karan
Enter height of the student 166
Enter name of student pinky

List of record
Height of the student : 1.7e+002
StudentName : pinky
Enter height of the student 181
Enter name of student jay

List of record
Height of the student : 1.8e+002
StudentName : jay
Process returned 0 (0x0)   execution time : 36.107 s
Press any key to continue.
```

Conclusion: From this program, I learnt about handling the files , various functions for Manipulation of File Pointers , File Modes in which operate and their working.